

High lift movement in a storage rack system

Pierre Stephan Baard



UNIVERSITEIT
iYUNIVESITHI
STELLENBOSCH
UNIVERSITY

100
1918 · 2018

Thesis presented in fulfilment of the requirements for the degree of
Master of Commerce (Operations Research)
in the Faculty of Economic and Management Sciences at Stellenbosch University

Supervisor: Prof SE Visagie

March 2018

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

March 2018

Abstract

With the growing competitive global economy new techniques need to be adopted for distribution of goods. Companies are searching for efficient distribution methods to improve customer service. Due to an increasing demand, distribution centres (DC) must be more versatile to keep up with a higher flow of goods.

Working closely with South-Africa's largest single brand retail chain, PEP, it is determined that there is a bottle neck in the pallet storage rack system at their Durban DC. With PEP's current system the rate at which pallets are brought into the storage rack system exceeds that at which pallets can be removed from the storage rack system. This difference, in the storage and retrieval rate of pallets, means that the DC needs large temporary storage areas for pallets waiting to be stored. Furthermore, the slow rate at which pallets are retrieved from the storage rack system slows all processes down the supply chain.

This study aims to find alternative movement logics, slotting configurations and job sequences for storing and removing pallets from the storage rack system in an attempt to increase the rate at which pallets can be stored and retrieved. Five high-lift movement logics are compared namely, **same side queueing without paired jobs**, **opposite side queueing without paired jobs**, **same side queueing with paired jobs**, **opposite side queueing with paired jobs** and **cyclic queueing with paired jobs**. The effect of changing the sequence of jobs are also analysed for each high-lift movement logic.

The job sequence is determined using heuristics that strive to find a good sequence. Four tabu search meta-heuristics are used that respectively sequence jobs by randomly swapping jobs, systematically swapping jobs from the top, finding the worst current job pairing and improving it and lastly finding the best current job pairing and improving it. The Hungarian method is also introduced in an attempt to find a better job sequence. Additionally, the pallet slotting problem is investigated to find the best storage rack placement configuration based on pallet replenishment cycles. The configurations considered are column pallet slotting (fastest moving inventory (FMI) closest to the picking line), row pallet slotting (FMI closest to the ground) and a stepping configuration with the FMI on the lowest step.

The movement logics, slotting configurations and job sequences are incorporated into a DC simulation program, coded in Python 3.4. Using data received from PEP's warehouse management software 10 consecutive shifts are selected for analysis. By simulating different combinations it is possible to find the one that reduces the process time the most. It is found that by changing PEP's current system to one that utilises same side queueing with job pairing the total cumulative completion time is reduced by approximately 39 hours and the distance travelled by high lifts by 45.27 km. This is further improved by adding the row pallet slotting configuration and a job sequencing algorithm. Using this combination reduces the total cumulative completion time by 43 hours.

Opsomming

Met die groeiende globale ekonomiese mededinging moet nuwe tegnieke aangewend word vir die distribusie van goedere. Maatskappye soek na doeltreffende distribusie metodes om kliëntediens te verbeter. As gevolg van die toename in vraag moet distribusiesentrums (DS) meer veelsydig word om trend te hou met die vloeï van goedere.

Deur samewerking met Suid-Afrika se grootste enkele handelsmerkwinkel, PEP, is daar vasgestel dat daar 'n bottelnek vorm in die stoorrakstelsel by hulle Durban DS. Met PEP se huidige stelsel is die tempo waarteen pallette in die stoorstelsel ingevoer word vinniger as wat pallette uit die stoorstelsel verwyder kan word. Hierdie verskil, in die stoor- en verwyderingstempo van pallette, beteken dat die DS gebruik moet maak van groot tydelike bergingsareas vir pallette wat wag om gestoor te word. Verder vertraag die stadige tempo waarteen pallette uit die stoorrekstelsel gehaal word, alle prosesse wat volg in die voorsieningsketting.

Die doel van die studie is om alternatiewe bewegingslogikas, pallet stooringskonfigurasies en volgordes te vind vir die berging en verwydering van pallette, uit die stoorrakstelsel. Dit word gedoen in 'n poging om die tempo te verhoog waarteen pallette gestoor en verwyder word. Vyf hyskraan bewegingslogikas word vergelyk, naamlik `same side queueing without paired jobs`, `opposite side queueing without paired jobs`, `same side queueing with paired jobs`, `opposite side queueing with paired jobs` and `cyclic queueing with paired jobs`. Terselfdertyd word 'n analise goed op die effek wat die verandering aan take se volgorde het, vir elke bewegingslogika.

Die volgorde van take word bepaal deur gebruik te maak van heuristieke wat probeer om die beste moontlike volgorde te vind. Vier tabusoektogte word gebruik wat onderskeidelik die volgorde verander deur lukraak take om te ruil, stelselmatig take om te ruil deur bo-aan die werkslys te begin, die slegste huidige werksparing te kies en te verbeter en laastens die beste huidige werksparing te vind en te verbeter. Die Hongaarse metode word ook gebruik in 'n poging om 'n beter werkreeks volgorde te vind. Daarbenewens word stoorrakkonfigurasies ondersoek om die beste stoorplek vir pallette te vind. Die konfigurasies wat in ag geneem word, is die kolompallet-konfigurasies (vinnigste bewegende voorraad (VBV) naaste aan die uitsoeklyne), rypalet-konfigurasies (VBV naaste aan die grond) en 'n trapkonfigurasie met die VBV op die laagste trap.

Die bewegingslogika, stoorrak-konfigurasies en taakvolgorde word gebruik in 'n DS simulatie, wat in Python 3.4 geprogrammeer is. Deur gebruik te maak van data wat vanaf PEP se pakhuisbestuursprogrammatuur ontvang is, word 10 opeenvolgende skofte vir analise gebruik. Deur verskillende kombinasies te simuleer is dit moontlik om vas te stel watter kombinasie die prosesseringstyd die meeste verminder. Die totale kumulatiewe voltooiingstyd verminder met ongeveer 39 ure en die afstand wat die hyskrane aflê verkort met 45.27 km. Dit word verder verbeter deur die rypallet-konfigurasies en 'n taakvolgordealgoritme by te voeg. Deur hierdie kombinasie te gebruik, verminder die totale kumulatiewe voltooiingstyd met ongeveer 43 ure.

Table of Contents

List of Figures	ix
List of Tables	xiii
List of Algorithms	xvii
List of Acronyms	xix
1 Introduction	1
1.1 Background on supply chains	1
1.2 Introduction to distribution centres	2
1.3 Types of DCs	4
1.4 Typical operations in a DC	6
1.5 Problem Description	8
1.6 Problem background	8
1.7 A short history of PEP	8
1.8 PEP's DC in Durban	9
1.8.1 Movement equipment	9
1.8.2 DC Layout	10
1.9 Scope and objectives	15
1.10 Thesis layout	16
2 Alternative logics and algorithms	17
2.1 High lift movement logics	17
2.1.1 Single job with opposite side queueing (OS)	18
2.1.2 Single job with same side queueing (SS)	19
2.1.3 Paired jobs with same side queueing (SSP)	20
2.1.4 Paired jobs with opposite side queueing (OSP)	21
2.1.5 Paired jobs with cyclic queueing (CS)	22

2.2	Job pairing sequence	23
2.3	Pallet slotting problem	26
2.3.1	Pallet classification	28
2.3.2	Replenishment cycle pallet slotting	29
3	Data	31
3.1	Warehouse management software	31
3.2	Raw data	32
3.3	Data validation	33
3.4	High lift movement time data	35
3.4.1	Storing and retrieving times	36
3.4.2	Analysis of the high lift travel times	37
3.5	Finding travel times	43
3.5.1	Travel times between rack locations and queues	43
3.5.2	Travel times between two rack locations	45
3.6	Job list generation	48
3.7	High lift assignment	49
3.8	RC Value assignment	50
4	Simulation model	53
4.1	Assumptions	53
4.2	Simulation model	54
4.3	Verification and validation	58
4.3.1	Verification	58
4.3.2	Validation	59
4.3.3	Output validation	61
4.4	Implementation	64
5	Simulation results	65
5.1	Sensitivity analysis	65
5.1.1	Movement logics analysis	66
5.1.2	Sequencing algorithm analysis	71
5.1.3	Slotting algorithm analysis	75
5.2	Historical data analysis (PEP)	77
5.2.1	Job split	77
5.2.2	High lift count	79
5.2.3	Movement logic and algorithm comparison	80

Table of Contents	vii
5.2.4 Historical completion time and distance	85
6 Conclusion	91
6.1 Recommendations	92
6.2 Future work	93
List of References	97
Appendices	99
A Algorithms	99
A.1 Movement logics examples	99
A.1.1 Single pallet movement types	99
A.1.2 Single pallet with opposite side queueing	100
A.1.3 Single pallet with same side queueing	100
A.1.4 Paired pallet movement logics	101
A.1.5 Paired jobs with same side queuing	101
A.1.6 Paired pallets with opposite side queuing	102
A.1.7 Paired pallets with cyclic queueing	103
A.2 Tabu search meta heuristics: Worst pairing Tabu search	103
B Tables and graphs	105
B.1 Data	105
B.1.1 Expected travel times of the high lifts	105
B.1.2 Regression models	105
B.1.3 Relocation travel times	108
B.1.4 Job list generation	110
C Decision support system	111
D Results	119
D.1 Movement logics analysis (Ratio = 25, 50, 75)	119
D.2 Movement logics analysis (Ratio = 0, 100)	120
D.3 Shift 2 analysis	120

List of Figures

1.1	The flow of product through a supply chain	1
1.2	The components of a supply chain.	2
1.3	A supply chain without a DC.	3
1.4	A supply chain with a DC.	3
1.5	An example of the floor storage in a common retail DC.	4
1.6	A typical catalogue fulfilment DC.	5
1.7	A typical perishable DC.	6
1.8	General product flow through PEP's Durban DC	9
1.9	Example of a pallet jack.	10
1.10	An example of the high lifts used by PEP.	11
1.11	The layout of PEP's DC in Durban.	11
1.12	The goods receiving area in PEP's DC in Durban.	12
1.13	The order picking area of PEP's DC in Durban.	12
1.14	The storage rack area of PEP's DC in Durban.	13
1.15	An example of the queues in the storage rack areas.	15
2.1	A single job system with opposite end queueing.	18
2.2	Single job system with same side queueing.	19
2.3	Paired job system with same end queueing.	20
2.4	Paired job system with opposite end queueing.	21
2.5	Paired job system with cyclic queueing.	22
2.6	Most convenient storage locations for pallets based on their replenishment cycle.	28
2.7	Examples of different pallet slotting configurations.	29
2.8	Shared storage areas applied to a pallet slotting configuration.	30
3.1	Historical data set's travel times after phase 1 clean up.	39
3.2	Total number of available storage times stamps.	40
3.3	Historical data set's travel times after phase 2 clean up.	42

3.4	Frequency graph for rack location 3012.	44
3.5	Process of generating travel times using each storage rack location's distribution.	44
3.6	Estimated average travel times between rack locations.	45
3.7	Travel time distribution focussing independently on the horizontal and vertical travel times receptively.	46
4.1	Process flow of the simulation model.	54
4.2	General flow and workings if the simulation model.	55
4.3	Comparison between the historical and estimated travel times for the retrieval jobs performed on rack location 3012	63
4.4	Number of simulations needed for results to converge.	64
5.1	General configuration of a storage rack.	65
5.2	Total movement distance for 1000 jobs with different configurations.	67
5.3	Breakdown of each movement logic's horisontal and vertical movement distances.	68
5.4	Average simulated completion time for different movement type configurations.	69
5.5	Breakdown of each movement logic's dead-heading and occupied time.	70
5.6	Average simulated completion time for different movement type configurations if the aisle length dead-heading time is ignored.	71
5.7	A comparison of adding a randomness function to the different sequencing algorithms.	72
5.8	Example illustrating the importance of job sequencing.	73
5.9	Total movement distance when applying a sequencing algorithm.	74
5.10	Total completion time when applying a sequencing algorithm.	75
5.11	The effect of a pallet's placement location on the distance when utilising movement logics that pait jobs.	77
5.12	Amount of storage and retrieval jobs performed in each rack for shift 2.	78
5.13	Seleceted pallet slotting configuration.	83
5.14	Cumulative completion distance for the simulated job list.	85
5.15	Cumulative completion time for the simulated job list.	87
5.16	Comparison of the average number of jobs performed per hour between the historical data and using the SSP movement logic.	88
5.17	Cumulative completion distance for the simulated job list using the row pallet slotting placement algorithm and five different movement logics.	88
5.18	Cumulative completion distance for the simulated job list using the row pallet slotting placement algorithm and five different movement logics.	89
A.1	General storage rack used in the movement logic examples.	99
B.1	Graphic representation of the travel times to store a pallet	105

B.2	Visual standardised comparison between the historical data and the estimated data.	107
C.1	GUI – Main menu of the simulation model’s decision support system.	112
C.2	GUI – Settings menu when data is generated for the simulation model	112
C.3	GUI – Main menu after the simulation is started.	113
C.4	GUI – Movement logics under the Examples tab.	114
C.5	GUI – simulating different movement logics under the ”Examples” tab.	115
C.6	GUI – The placement algorithms under the ”Examples” tabs.	115
C.7	GUI – Rack painter underneath the ”Editor” tab.	116
C.8	GUI – The rack display underneath the ”Editor” tab.	117
C.9	GUI – Settings available when using the rack display.	118

List of Tables

3.1	Format of the data received from PEP	32
3.2	Phase 1 clean up process.	33
3.3	Phase 2 clean up process.	34
3.4	Three types of travel times present in the data set	36
3.5	Summarised high lift pick up and put down times.	37
3.6	R^2 values for the median travel times using the historical data.	40
3.7	R^2 values for the average travel times using the historical data.	41
3.8	P-values for the median travel times using the cleaned historical data.	43
3.9	Calculated p-values for each cell location for known distribution using Minitab 17.	43
3.10	p-Values btained when testing the null hypothesis between the estimated data and the historical data's regression lines	47
3.11	Coefficient of determination, R^2 , between estimated and historical travel times.	47
3.12	Break down of the number of clean ups performed on each month.	48
3.13	Number of anomalies per group for November 2015.	49
3.14	Time interval for each shift.	49
3.15	High lift assignment	49
3.16	Example extracted from 14-Nov-2015 showing storage jobs with undefined RC values in the historical job list.	50
3.17	Number of storage jobs that each have a RC value assigned to them.	50
3.18	Number of pallets, without a RC values assigned, stored per time catagory.	51
3.19	Number of jobs that have each RC value assigned to them after assigining new RC values.	51
4.1	Storage rack aisle pairings.	57
4.2	Exaple of how processe communicate with each other.	57
4.3	Validation of estimated travel times.	62
4.4	Null hypothesis p-values for simulated and historical travel times' distributions.	62
4.5	Validation of estimated travel distances.	63

5.1	General format of a job list constructed from historical data.	66
5.2	Summary of difference percentage in average completion times for the differnt movement type configurations.	69
5.3	Average simulated completion time for job lists only consisting of storage or retrieval jobs.	70
5.4	Average time required by each sequencing algorithm to converge to the best know solution	73
5.5	Average number of jobs stored before there are no more storage locations available for each pallet slotting configuration.	76
5.6	Average total distance each movement logic required to complete all jobs with different slotting configurations	76
5.7	Number of storage and retrieval for each shift with their difference percentages. .	78
5.8	Number of distributions present in the categories front and back for each storage rack	79
5.9	The effect adding more high lifts have on the average cumulative completion time and distance	79
5.10	Aisle break down of job types for shift 2	80
5.11	Summary of average completion times, in seconds, and completion distances, in rack locations, when applying the five movement logics to the jobs performed in shift 2 to for each aisle.	81
5.12	Summary of average completion times, in seconds, and completion distances, in rack locations, when applying the five movement logics in unison with a sequencing algorithm to the jobs performed in shift 2 to for each aisle.	82
5.13	Summary of average completion times, in seconds, and completion distances, in rack locations, when applying the five movement logics in unison with a sequencing and slotting algorithms to the jobs performed in shift 2 for each aisle.	84
5.14	Example of how the historical data does not follow the flow described by PEP. .	86
5.15	The percentage each movement logic improves on the distance of simulating the job list exactly.	86
5.16	The percentage each movement logic improves on the time of simulating the job list exactly.	87
5.17	The effect an empty and 90% full storage rack system has on the differnt movement logics.	89
5.18	The effect an empty and 90% full storage rack system has on the differnt movement logics in terms of completion time.	90
A.1	Generated job list for the movement logics that moves one pallet per job.	99
A.2	Single pallet with opposite side queueing example results.	100
A.3	Single pallet with same side queueing example results.	101
A.4	Generated job list for the movement logics that moves one pallet per job.	101
A.5	Paired jobs with same side queueing example results	102

A.6	Paired pallets with opposite side queuing example results	102
A.7	Paired pallets with cyclic queueing example results	103
B.1	Correlation coefficient equations for each linear regression line of the data sets. .	106
B.2	Standardised data sets.	106
B.3	Summary of the equations used to calculate the R^2 value for the minimum, average and maximum travel times between rack location.	108
B.4	The maximum, minimum and median travel times, in seconds, to each rack location.	109
B.5	Number of jobs completed each day including the number of anomalies for November 2015	110
D.1	Average simulated completion times for different movement type configurations using a job ration of 25.	119
D.2	Average simulated completion times for different movement type configurations using a job ration of 50.	119
D.3	Average simulated completion times for different movement type configurations using a job ration of 75.	119
D.4	Summary of average simulated completion times for different movement type configurations.	120
D.5	Average simulated completion time for job lists only consisting of storage or retrieval jobs.	120
D.6	Average simulated completion time for job lists only consisting of storage or retrieval jobs.	120
D.7	The improvement percentages achieved when applying the sequencer algorithm to the different movement logics for average completion time and distance.	121
D.8	The improvement percentages achieved when applying the sequencer and slotting algorithms to the different movement logics for average completion time and distance.	121
D.9	The average travel time, in seconds, a high lifts takes to move from one rack location to another.	122
D.10	Summary of the cumulative travel distances using row pallet slotting placement algorithm in conjunction with each movement logic split into the occupied, dead-heading and total distance.	122
D.11	Summary of the cumulative travel distances using row pallet slotting placement algorithm in conjunction with each movement logic split into the occupied, dead-heading and total distance. While simulating each movement logic the storage rack system is filled to 90% capacity.	123

List of Algorithms

1	General Tabu search extract	24
2	Random Tabu Search	24
3	Pairing from top Tabu Search	25
4	Best pairing Tabu search	27
5	Worst pairing Tabu search	104

List of Acronyms

DC: Distribution centre

RC: Replenishment cycle

OS: Opposite side queueing without job pairing

SS: Same side queueing without job pairing

OSP: Opposite side queueing with job pairing

SSP: Same side queueing with job pairing

CS: Cyclic queueing with job pairing

FMI: Fastest moving inventory

WMS: Warehouse management software

SKU: Stock keeping unit

LRL: Linear regression line

MVR: Multi-variable regression

CHAPTER 1

Introduction

Fierce competition in today's trading sector have forced businesses to invest and focus on improving their operations to become more competitive. Due to the continuing advances made in communication and transportation technologies, more companies are competing on the global market. Companies are forced to optimise their operations to remain competitive in this environment.

1.1 Background on supply chains

A supply chain describes a network of processes, people, information and resources participating in the procedure of moving a product or service from its natural resources to a customer. Ayers [1] identifies several components that form a supply chain. It includes manufacturing, suppliers, distribution, retailers, customers and logistics. The typical flow of products in a supply chain is illustrated in Figure 1.1. Harvesting natural resources often constitutes the start of supply chains. Gathered resources are processed, refined and manufactured into goods. The goods are sent to distribution centers (DCs), where they may be processed further and sent to specified stores. The consumer or end user is then able to access the final product or service from the stores.

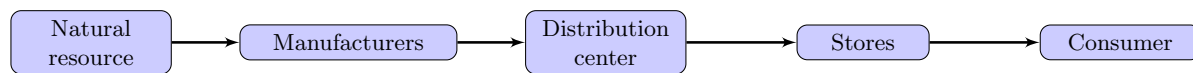


Figure 1.1: Product flow through a supply chain.

The interaction between each of the components of a supply chain is illustrated in Figure 1.2. Raw materials are extracted and sold to suppliers. The supplier dispatches the raw materials to manufactures where it is processed and refined into finished goods. Retailers buy the finished products from manufactures in large quantities. The finished product is shipped from the manufacture to the DC and processed further. DCs typically function as a middleman between manufacturers and retailers. From the DC the products are shipped to retailers in more manageable order sizes. The consumer is the final recipient of the finished product. Logistics comprises of the management of how goods need to flow between the different parts of the supply chain in order to meet all the demands. A supply chain is not only confined to physical products but can be the flow of information, money or knowledge [1].

The effectiveness of supply chain plays a crucial part for most companies in lowering expenses with efficient supply chains usually translating to lower costs [1]. Benefits of a well designed,

operated and maintained supply chains include lower operation costs, improved customer satisfaction and a competitive edge through fast and accurate product delivery [2]. Therefore, it is beneficial to attempt to optimise operations that govern product flow in supply chains including sourcing, manufacturing and shipping.

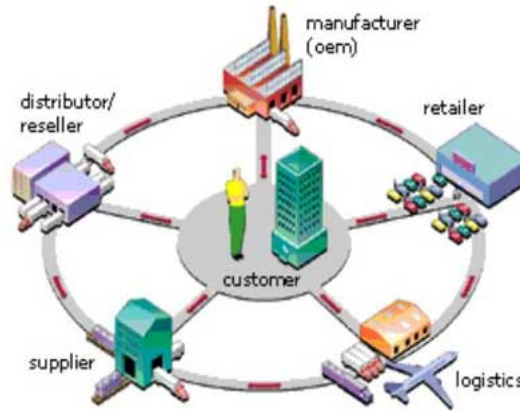


Figure 1.2: A schematic representation of the components of a supply chain. Source: [27]

1.2 Introduction to distribution centres

Tompkins [40] states that the five functionalities DCs serve are: stockpiling, product mixing, consolidation, shipping and improving customer satisfaction.

A DC is often a large commercial building or plot where goods are stockpiled, catalogued and reorganised. DCs are often used by manufacturers, importers, exporters, wholesalers or transportation businesses. Stock that is commonly stored in a DC include raw materials, packing material, spare parts, components and finished products.

Due to stock generally being stored in batches, it is not uncommon for a DC to have an abundance of forklifts and cranes moving stock around. Some DCs are equipped with loading docks to directly load and/or unload stock onto trucks, rail roads, sea ports or airports. In the past most DCs needed to be situated in industrial areas near major shipping ports in order to keep efficiency high. The DCs would have had large work forces handling all the different equipment but with today's technology it is not uncommon for DCs to be completely automated, depending on the wealth of the company and the type of stock [45].

DCs plays a vital part in managing the distribution process, lowering transport cost and adding value to the supply chain. Product supply does not change at the same rate as demand which establishes a demand and supply imbalance [2]. Consumer goods industries trade in products such as clothes, food and entertainment equipment and are greatly affected by seasonality when compared to durable goods industries that trade in products such as electronics and motor vehicles. It is especially important for these more volatile industries to manage demand changes with safety stock to increase customer satisfaction by meeting demand. DCs allow for any overflow from the manufacturer to be stored or to store safety stock to account for stochastic demand changes or seasonality of stock.

Space in a store is much more expensive than industrial space. It is thus inefficient to store extra stock in a store's storage space. To prevent the unnecessary wastage of money an off-site storage is needed that is capable of storing the excess stock, usually in a DC. This DC, however, needs to be close enough as not to increase the transportation cost and to allow the store to

easily access it when product demand is not reached. A DC may also serve as storage space for stock that were not sold during its selling season.

A further benefit utilising DCs is the reduction in transportation costs due to economies of scale. When stock is transported via ship, air or train a fixed cost is allocated to the means of transport [2]. If a single store is to order stock directly from manufacturers, the order would probably not be big enough to order stock on a scale capable of filling an entire cargo container. The store would, however, still receive the entire fixed shipping cost even though not all the space is utilised. DCs usually have enough storage space to be capable of ordering full containers when ordering stock from manufacturers, leading to less money wasted on ordering half shipments.

Utilising a DC can further reduce transportation cost by decreasing the number of transport routes. Figure 1.3 illustrates a supply chain that does not utilise a DC. The depicted supply chain consists of three manufacturers (M) and three stores (S). The total number of transportation routes utilised would, in this instance be nine.

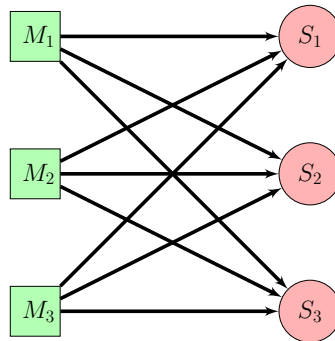


Figure 1.3: A schematic representation of a supply chain with three manufacturers, represented as squares, and three stores, represented as circles. The connecting arrows indicate transportation routes.

By introducing a DC into the supply chain, it changes the connection configuration between manufactures and stores as shown in Figure 1.4. Using three manufactures (M) and three stores (S) the total number of transport routes are reduced to six. These shipments would be more cost effective due to shortened travel distances and the higher possibility of filling shipments to capacity.

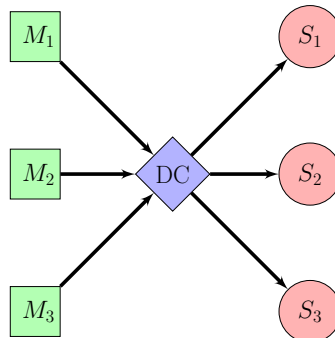


Figure 1.4: A schematic representation of a supply chain with three manufacturers, represented as squares, three stores, represented as circles and a DC, represented by the diamond. The connecting arrows indicate transportation routes.

If manufacturers produce specific products in different production sites, DCs may be used as product mixing locations. Different products can be mixed together at the DC to produce one

product. This is helpful in cases where customers order large quantities of a single product at a time. Consolidation is thus also an important function of a DC. It could be one of the uses of a DC to receive, store and combine various products into single batches to fill orders.

It is also possible to label and price individual products directly at the DC. It is less expensive for this to be done in a few DCs where the product is already handled, instead of in a multitude of stores where it can cause staff to neglect customers.

1.3 Types of DCs

Bartholdi [2] categorises DCs into four types, namely retail DCs, service parts DCs, catalogue fulfilment or e-commerce DCs and perishables goods DCs depending on the type of customers they serve.

A retail DC's main customers are retail stores and typically sends shipments to these stores on a regular basis. Retail DCs are often used in consumer markets and typically handle small to medium sized products. Each order that is sent can comprise of hundreds to thousands of items at a time. The type of stock and volume that is sent by the DC is based on the demand of customers and the marketing strategy/promotions applied to products. It is possible for a single DC to have many retail stores to which it distribute stock, which often results in large stock flows [2]. Retail DCs often store safety stock to compensate for abrupt changes in demand. Retail DCs need enough storage space to accommodate long term safety stock and slow moving stock. Fast moving stock are generally stored on floor storage spaces for easy and quick access. Due to the nature of these DCs they are usually situated in large industrial buildings. Retail DCs tend to utilise storage racks as illustrated in Figure 1.5 to utilise all of the vertical space.



Figure 1.5: A photo of the floor space of a common retail DC trading in goods like clothes, electronic or entertainment equipment and furniture or appliances. Batches of cartons are stored on a small floor storage area. Storage racks used for storing slow moving products are visible in the background of the photo. Source: [43]

Service parts DCs are the most challenging to manage [2]. They store expensive capital equip-

ment like medical equipment, auto mobile components or computer components. These DCs are capable of storing and processing large volumes of stock. Working only with large quantities allows for easier statistical modelling of the overall inventory levels in the DC. However, the demand for any one of these products can be relatively small, making it hard to predict the demand for individual products. This means large volumes of safety stock must be stored due to the difficulty in predicting the demand of a specific product. The lead times from suppliers are also usually long when trading in capital equipment. This could lead, in some cases, to the volume of safety stock being as much as the cycle stock (stock that is available for normal demand in a given time frame) [2]. Service parts DCs usually only manages two types of orders namely stock orders, where stores replenish their display shelves, and emergency orders, when a store unsuspectingly requires a few special items to execute its services, like repairing a piece of capital equipment [2].

Catalogue fulfilment or e-commerce DCs usually receive small orders (one or two items) from individual customers via phone, fax, or the internet. There are many small orders from many different customers and must be distributed as soon as possible after the order is placed. With this type of DCs fast delivery times are vital to keep customers satisfied. Further stock need to be distributed quickly to ensure that the DC's storage capacity is not exceeded [2]. The rate at which orders are received often changes over a week or day, depending on when customers are online — such as after work or weekends. These types of DCs must manage the spikes of orders and bundle the seasonal outflows. This is typically achieved by promoting stock by means of special prices at certain times so that the DC can ship a bulk order. Time delays between purchase time and actual shipping times allow for the use of strategic marketing to reduce shipping cost. Usually these types of DCs have set shipping times. All orders placed between shipping times are shipped together. If there is enough time between shipments a DC might promote a lower price on certain stock in an attempt to fill delivery vehicles to waste as little space as possible. These types of DCs have very large floor storage spaces. Figure 1.6 shows the use of large floor storage areas which allows batches of stock to be easily accessible to decrease handling times.



Figure 1.6: A photo of a typical catalogue fulfilment DC, which shows the use of large floor storage areas. Small batches of stock are spread out across the floor to increase accessibility. Source: [46]

Perishable goods DCs hold goods with short life spans such as fresh food, flowers, vaccines, or any other product that needs to be refrigerated [2]. The stock stored in this type of DC moves fast relative to the other types of DCs and it is often only in the DC for a few hours before being distributed. Utilising all the available space is a high priority for these DCs as refrigeration space is expensive. Products get distributed according to a FIFO (first in first out) or FEFO (first expired first out) system.



Figure 1.7: A photo of a typical perishable DC. Cooling equipment can be seen at the back keeping the stock at the correct temperature. source: [3]

1.4 Typical operations in a DC

DCs often distinguish between two types of processes namely inbound and outbound. Inbound processes consist of all the operations performed on stock from its arrival, at the DC, to being stored. The outbound processes consist of all the operations performed on stock from when it is retrieved from storage to being shipped out of the DC.

One of the primary operations performed in a DC includes reorganising and repacking stock, collectively referred to as the order pick operation. Products typically arrive at the DC on a large scale. Inside the DC large batches of stock keeping units (SKUs) are broken down into smaller units (individual units, cartons or packs) and distributed. A greater handling cost is associated with smaller handling units [2].

Although there are many different types of DCs they all tend to have the same general flow of stock. Lahmar [17] explains that the general product flow can be categorised into four broad categories. They are product receiving, product put-away, picking lines and assembly and lastly shipping.

Receiving

Receiving is generally the point at which the responsibility of the stock shifts from manufacturer to that of the DC [40]. Once stock arrives at the DC, usually in containers on trucks, it is

unloaded and staged for put-away. During the unloading process stock is scanned, labelled and reorganized. A quality control check is typically performed at this point to ensure the correct quantity of stock has arrived and that the quality is acceptable. Products sometimes arrive already packed in batches. These batches need to be broken down into smaller units based on how and where they need to be stored in the DC. If different products arrive in the same delivery vehicle the different products are usually grouped separately onto different pallets.

Put-away

After stock is received and checked, it is put-away into the storage areas. A vital decision is made during put-away, namely to which storage area each pallet is assigned. The storage locations of pallets influence the speed and cost to retrieve the stock at a later time. This requires good management of pallets and the availability of storing locations. It is important to know what locations are available for storage, the size of these locations and the weight/size it can take. After a pallet is stored its location is noted to ensure its location is known for when it must be retrieved.

Order picking and assembly

According to Frazelle [11] ordering picking is a basic service that most DCs provide. It is also the operation that most DCs base their design around. Order picking is the process of removing individual product items (or groups of products) from storage to make up individual orders of various products. This is often a costly and time intense process as it is usually performed by humans. What makes it even more time consuming is the fact that products are being handled in their smallest possible size, resulting in a large number of units being handled during picking. Order picking can make up about 55% of the DC's operational cost [2].

Order picking is typically done by manual labour. An order is either entirely picked by a single worker, or by many workers. The strategy and number of workers (pickers) needed to pick an order depends on a number of factors. The primary contributors are the deadline of an order and the rate at which products should be picked. A single picker might take longer to complete an order but avoids the complications of coordinating multiple workers and combining the different parts of an order.

In some cases after a product is picked it is batched with the rest of the order. This is a labour-intensive operation due to each item, in an order, being handled individually. Since each piece of the order gets handled it is usually checked to ensure the order is correct and accurate. Accuracy of an order is important, to maintain a competitive advantage [2]. When packing products it is essential that it is packed as small as possible to minimise the transportation cost.

Shipping

The shipping operation handles all consolidated orders. Picked products are packaged together in batches resulting in fewer units to be handled. These batches are packed into trucks according to the destination of the products. Trucks are usually packed in reverse order of delivery [2]. This method could result in longer packing times and double handling since trucks are packed as close to capacity as possible.

1.5 Problem Description

Many retailers in South Africa make use of DCs. Knowing the importance of a DC with regards to efficiency and profitability it is fundamental that this link in the supply chain is as effective as possible. Due to the intricacy of DCs they are frequently managed inefficiently. This could be due to disorganised usage of storage space, ineffective picking processes or ineffective stock slotting. Although it is common for retailers to make use of DCs it is uncommon for them to be managed the same.

Because of the vast differences between each retailer's management of their DCs a single retailer is selected. The clothing retailer PEP stores Ltd (PEP) is selected and their DC in Durban is considered.

The management of PEP's Durban DC believes that congestion exists within their storage rack area. This is due to stock requiring a higher rate of flow into and from the storage rack area than is currently possible. As a result of the low put-away rate, stock is stored in temporary storage areas in the receiving area for longer. As the temporary storage areas become full all preceding processes are slowed and becomes less efficient. When stock is removed from the storage rack area at a slower than required rate, all ensuing processes are halted as they are waiting for stock replenishment. This, in effect, means that the DC is unable to function at maximum efficiency. It is thus crucial for the operations in the storage racks to be improved to enable the DC to perform more efficiently.

Before any attempt can be made to increase the efficiency of the storage rack a proper understanding of all operations within PEP's Durban DC is required. More insight is also required regarding all operations executed within the storage racks. More info regarding PEP's Durban DC is discussed in the following section.

1.6 Problem background

PEP is the largest single brand retail chain in South Africa. PEP's main target market is people with a tight budget. PEP sells apparel but also trades in home-ware, fast moving consumer goods, cellular- and airtime products. PEP also offers services like cash-backs (customers can withdraw cash when making a debit card purchase), Capfin (for loans), funeral policies, cross-border money transfers, selected payments of bills, flash electricity tokens and more [31].

PEP's supply chain in Southern Africa consists of approximately 2 000 retail stores that are serviced by three DCs. The smallest DC is in Kuilsriver in the Western Cape, the largest is situated in Insando (Johannesburg) in the Gauteng province and the last one is located in Isipingo (Durban), KwaZulu-Natal. Although the Durban DC is not the largest in terms of floor space, it is still seen as PEP's main DC as it receives the majority of PEP's products from South Africa's main port in Durban. The focus of this study therefore shifts to the DC in Durban.

1.7 A short history of PEP

The first PEP store opened in 1965 in De Aar. It was started by Renier van Rooyen as a small discount store to cater for the very poor people in the Northern Cape. The store was one of the first shops in South Africa that catered to the poor community by providing quality clothing at very low prices [10]. They also provided an equal service for all races, allowing all customers to

touch and hold the merchandise before buying, unlike most shops where products were displayed behind a counter.

PEP expanded very quickly with its high quality low priced products. Since the late 1960s PEP had dominated the cheap retail sector for three decades [10]. During the 1990s PEP underwent a decline in market share and André Labuschaigne was appointed in 1998 to bring about a change to avoid closure. This change has led to PEP being the biggest single brand retailer in Africa today. PEP operates more than 2 000 retail stores in Southern Africa and employs more than 17 000 people [30].

1.8 PEP's DC in Durban

The conventional flow of stock through PEP's Durban DC is shown in Figure 1.8. Stock is received at the DC from different suppliers. As stock gets unloaded from trucks it is temporarily stored in the receiving area. Stock is then either moved to floor storage areas near the order picking area or they are stored in the storage racks. When the time arrives for a pallet to be removed from the storage racks it is moved to the order picking area. Orders are picked according to store order and moved to the shipping area where the stock is loaded into trucks and transported to retail stores.

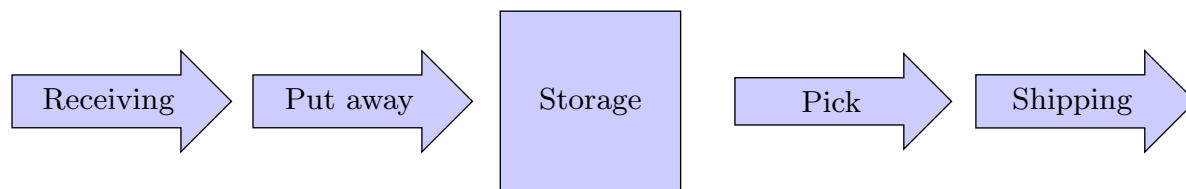


Figure 1.8: A schematic representation of the typical pallet's flow through a DC.

There are two main forms in which stock is moved in the DC. The first being batches of stock that are combined into pallets and the second being single cartons. Stock is moved throughout the DC using numerous forms of equipment.

1.8.1 Movement equipment

The movement equipment that operate within the DC can be categorised into three types, namely pallet jacks, picking forklifts and high lifts. Pallet jacks are the most basic and abundant pallet moving equipment used. Figure 1.9 shows that they are very agile equipment used to move one or two pallets at a time. In the DC there are two types of pallet jacks that are used namely manual pallets jacks and electric pallets jacks. The manual pallet jacks are operated by either being pushed or pulled in order to move around. The electric pallet jack has an electric motor that supplies the necessary torque to move pallets. Due to the size of the electric pallet jack they are mainly used to move pallets in areas with ample space. Pallet jacks are only able to store pallets on the ground level but are used throughout the DC to move pallets between the different areas as indicated in Figure 1.11.

The second type of equipment that operates within the DC are picking forklifts. These forklifts mainly serve the same purpose as the pallet jacks, moving pallets through the DC. They are, however, capable of lifting pallets and placing them on top of one another. This allows the DC to utilise more space. They also act as pickers for new stores by removing the stock from the storage areas that are not destined for the order picking area. When doing this they pick



Figure 1.9: A photo of the type of pallet jack used in the Durban DC to move pallets from one area to another. Source: [41].

individual cartons from specified pallets, giving a higher pick priority to pallets with fewer cartons remaining. As long as a single carton remains on a pallet that specific storage location stays unavailable. Pallets with fewer cartons are thus given priority by forklifts to clear pallets and make storage location available as fast as possible and not waste storage space.

High lifts, shown in Figure 1.10, are only found in the storage racks. These are the most important movement equipment operating in the storing racks and are responsible for the storing and retrieval of pallets from rack locations. Each high lift is equipped with a fork capable of rotating 180° , allowing for pallets to be pick up and placed next to and in front of the high lift. It also enables the high lifts to access both sides of an aisle in the storage racks. High lifts differ from conventional forklifts in the sense that they are able to reach far greater heights and that the high lift operator is elevated with the forks. Due to the purchasing and maintenance cost associated with high lifts the Durban DC only utilises five high lifts at any given time.

1.8.2 DC Layout

The arrows in Figure 1.11 indicate the typical flow of stock through the Durban DC. Stock is ordered from either local or foreign manufactures which are predominantly China and India. The stock typically arrives at the Durban DC six to ten months (including manufacturing and shipping time) after being ordered. Stock arrives at the DC, stored in cartons, by means of trucks. The cartons are unloaded and stored on pallets according to their characteristics. After a pallet is fully loaded the pallet is temporarily stored in the goods received area. From here the pallets are moved to either the storage racks, full carton area or order picking area.

Based on demand, pallets are removed from the storage rack area and moved to the order picking area. At the order picking area pallets from the storage racks are placed in temporary storage area and SKUs are picked according to order. Picked SKUs are packed into cartons and placed on conveyor belts moving the cartons to the shipping area. At the shipping area full pallets and picked cartons are grouped according to their end destinations. Groupings of stock are packed into trucks and transported to the respective retail stores.



Figure 1.10: A photo of the high lifts used in the storage racks. The forks of the high lift can rotate 180° and can thus access pallets on either side of an aisle in the storage racks. Source: [34]

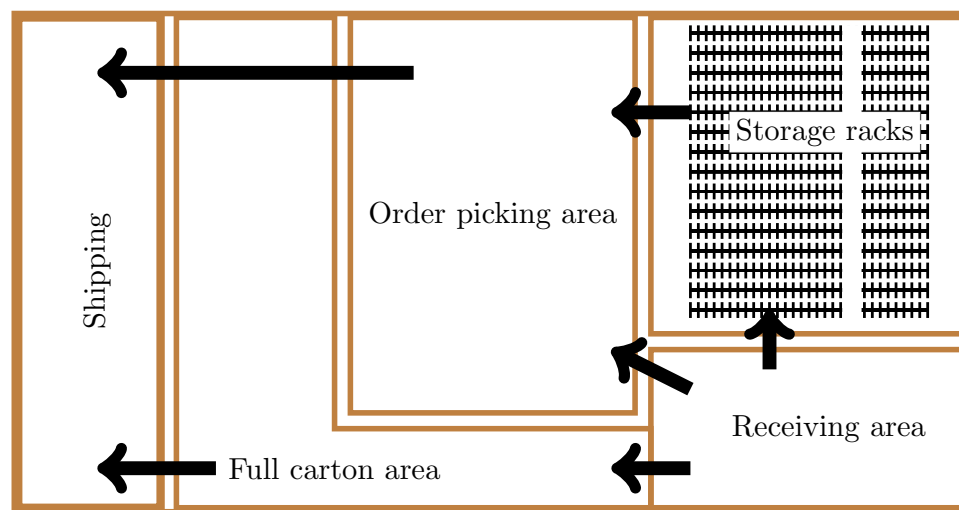


Figure 1.11: A schematic representation of the layout of the DC in Durban. The arrows indicate the flow of stock in the DC. The thin solid lines indicate the different zones in the DC.

Receiving area

Stock arrives at the DC in trucks from the Durban harbour or local manufacturers. As the trucks arrive at the DC they are queued in front of the receiving area. There are fifteen loading bays for delivery vehicles at the receiving area. The layout of the receiving area is shown in Figure 1.12. Loading bay 1 is for small/light vehicles. Loading bays 2 – 8 for large vehicles arriving from the harbour and loading bays 9 – 15 for trucks from local suppliers. The stock, which is typically stored in cartons, is offloaded from the trucks using conveyor belts, shown in Figure 1.12.

As the cartons move down the conveyor belt, out of the truck, randomly selected cartons are removed for quality check purposes while the rest are stacked on pallets. After a quality check is complete, on the removed carton, it is placed on the conveyor belt again. Stock is placed on

different pallets according to their specific characteristics. When a pallet is filled to capacity it is labelled for identification purposes. The label has all the necessary information regarding the pallet. The information on the label consists of the product info, location for storage, bar code and time of arrival. From the receiving area pallets are either moved to the full carton area, order picking area or to a temporary storage space, shown in Figure 1.12, (also referred to as the pick and drop area) using pallet jacks. From the pick and drop area pallets are moved to the storage rack area. The full carton area is a floor storage area where goods are stored. Full cartons refer to stock that is distributed as full cartons and this does not go to the picking lines.

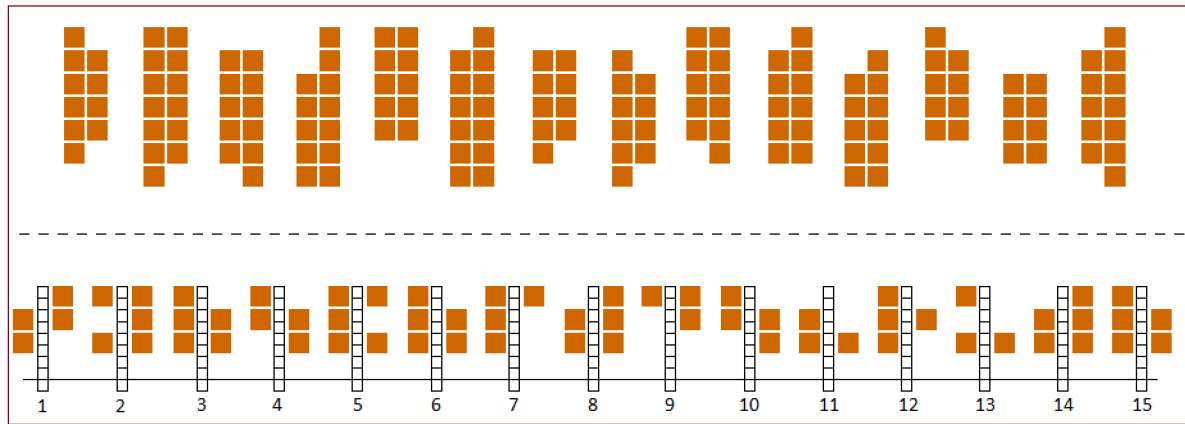


Figure 1.12: A schematic representation of the goods receiving area in the DC in Durban with fifteen loading bays. The area below the dashed line is where carton are off loaded from truck. The narrow dashed rectangles indicate the conveyor belts while the solid squares represents pallets that are packed. The area above the dashed line is the temporary storage space (pick and drop) in the goods received area.

Order picking area

The order picking area is divided into eight zones called picking lines. Shown in Figure 1.13, these picking lines comprise of 56 bin locations evenly spread around a conveyor belt with each bin location being occupied by a maximum of one pallet. In PEP's Durban DC picking lines make use of a picker-to-parts system. In this system pickers move in aisles along bins where stock has been placed. They then pick stock from the stored pallets according to placed orders.

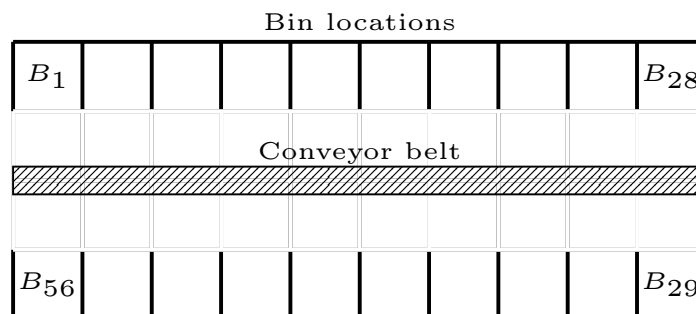


Figure 1.13: A schematic representation of the order picking area in PEP's DC in Durban.

Picking lines are built according to a collective of orders received from retail stores. Using pallet jacks the pallets containing the required SKUs are moved from the storage racks area to the

picking lines and placed in the pre-defined bin locations by forklifts. A picking line can only become active when all required pallets have been placed in the bins.

Pickers, usually eight working in each picking line, walk around the conveyor belt in a clockwise direction. A picker is limited to picking stock for only one order at a time. When a picker passes a bin containing stock relevant to their current order it is picked and the ordered quantity is placed in a carton with the rest of the picked stock. In the event that all SKUs have been picked from a pallet the pallet is replaced with one from storage. Once a picker completes an order the carton is placed on the conveyor belt and the picker is given a new order. From here the cartons are moved to the shipping area, passing through quality control first.

Storage racks

Pallets arrive at the storage racks by means of pallet jacks and are placed in storage queues at the end of the aisles. Pallets are moved within the storage rack aisles by means of high lifts. There are nine high lifts in the DC of which typically only five are operating during each working shift.

The storage rack area can be categorised into two sets of storage racks, S1 and S2, as shown in Figure 1.14. Both sets consist of 24 aisles with 48 storage racks six levels high. The set S1 in Figure 1.14, closest to the order picking area, consist of 32 rack locations on each level per rack while set S2 only has 20 rack locations per rack on each level. Set S1 typically stores stock with a faster replenishment cycle, up to 28 days, while S2 is used to store stock that will be stored for longer times. The majority of stock is stored in S1 and is usually 90% filled [23]. Aisle 1, Figure 1.14, is reserved for individual cartons that are returned from the order picking area or stock that is reserved for new stores. No pallets are stored in these aisles and cartons are picked using forklifts. The majority of pallets stored on the top level of each storage rack, level 6, are also used for new stores.

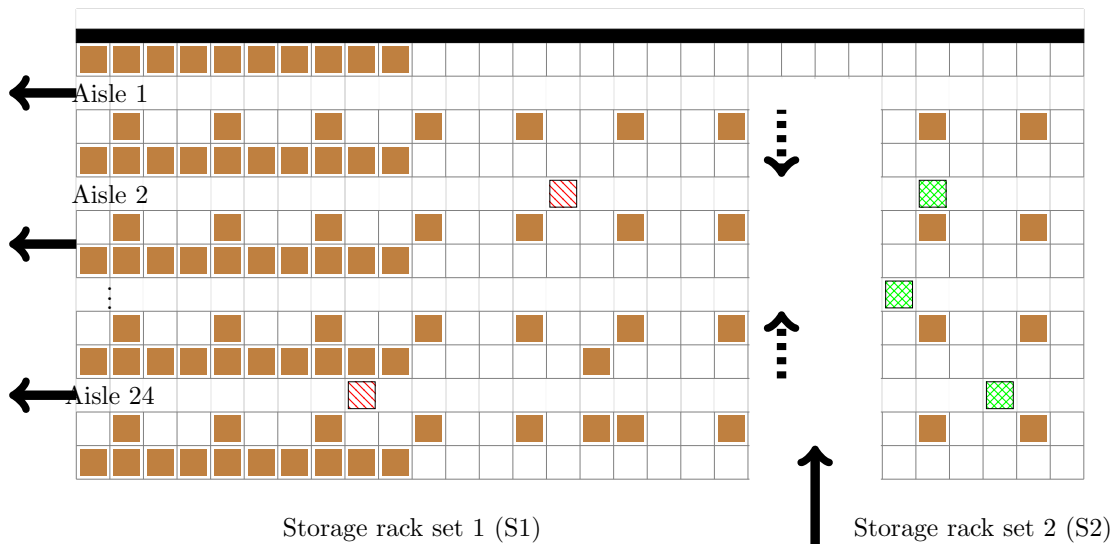


Figure 1.14: A schematic representation of the storage rack area in PEP's DC in Durban. The green cross hatch squares represent three high lifts doing put-away jobs and the red lined squares indicate two high lifts doing retrieving jobs. The black arrows show the general flow of pallets through the storage racks area. The small dashed arrows show the typical direction in which jobs are completed by the high lifts that store pallets. The thick black bar at the top represents the outer wall of the DC.

There are two primary types of jobs, storage and retrieval, which govern the processes of high lifts in the storage racks system. During a storage job a high lift removes a pallet from a storage queue and stores it in the assigned rack location for future retrieval. Pallets may stay stored for periods from less than a week to six months. Retrieval jobs originate when pallets are collected from their rack locations and placed in retrieval queues. During a typical working shift there are two high lifts busy with removing pallets and three busy storing pallets or the other way around — depending on the work balance.

There is a third job type performed by high lifts in the storage racks, namely relocation jobs. Relocation jobs do not happen often and no specific jobs are assigned in advance for them [23]. These jobs happen when a pallet is either stored in the wrong location or needs to be moved to a different location due to stock clean up.

During a work shift jobs are grouped according to type. These groups are then presented in batches, also referred to as job lists, and assigned to the high lift operators. It is common practice in PEP's Durban DC that a high lift operator is assigned a batch of the jobs that are of the same type — meaning all assigned jobs are either storage- or retrieval jobs.

At the beginning of each shift new job lists are printed and assigned to high lift operators. These printed job lists contain all pallets that need to be retrieved during the shift. Retrieval job batches are sorted, according to the storage rack number in which they are stored. High lifts that are busy processing a batch of retrieval jobs, move to the pallet location in the storage rack according to the job list. At arrival the pallet is removed and moved to the corresponding retrieval queue. The pallets are then moved from the retrieval queue to the order picking area by means of pallet jacks. This process is repeated until all the retrieval jobs in the job list are completed. High lift operators start on opposite ends of the storage racks, as depicted with the dashed arrows in Figure 1.14, working their way inward to the middle. When a high lift reaches the middle aisle a new job list is started. The DC uses a strict safety rule that states only one piece of movement equipment is allowed in a single aisle at any given time. High lift receives priority above the other movement equipment types. This is done to prevent two high lifts operator selecting the same job from the job list and possibly going into an aisle that is already occupied.

Pallet jacks supply the storage rack system with randomly chosen pallets from the pick and drop area. The pallet jacks place the pallets in the storage queues where high lifts are currently working. High lifts in the process of storing pallets, remove pallets from the storage queue and place them in their allocated rack locations. This process is repeated until all pallets in the storage queue are stored. After a queue is emptied the high lift moves to the next queue with pallets. This means that high lift operators who are assigned to do storage jobs are dependant on pallet jacks to supply them with jobs.

Pallets wait in storage queues before it is stored. After a pallet is retrieved it is placed in a retrieval queue. These queues are situated at the ends and on opposite sides of each aisle. The storage queues of both storage rack sets are situated between the two sets of racks. The retrieval queues of pallets removed from S1 are situated on the order picking area side of the storage rack system. The opposite side of storage rack set S2, this is the right side of the storage rack in Figure 1.14, is used as a storage area and does not have adequate space for queues to be formed. The retrieval queues for storage rack S2 are thus on the same end of the aisle as the storage queues.

The length of queues need to be kept as short as possible to ensure adequate moving space between aisles. Ted Moodley [23] mentioned that the maximum length a single queue is capable of reaching before it becomes a problem is four pallets. Additionally, the maximum combined queue length of opposite queues are four pallets as well. Figure 1.15 shows that when queues

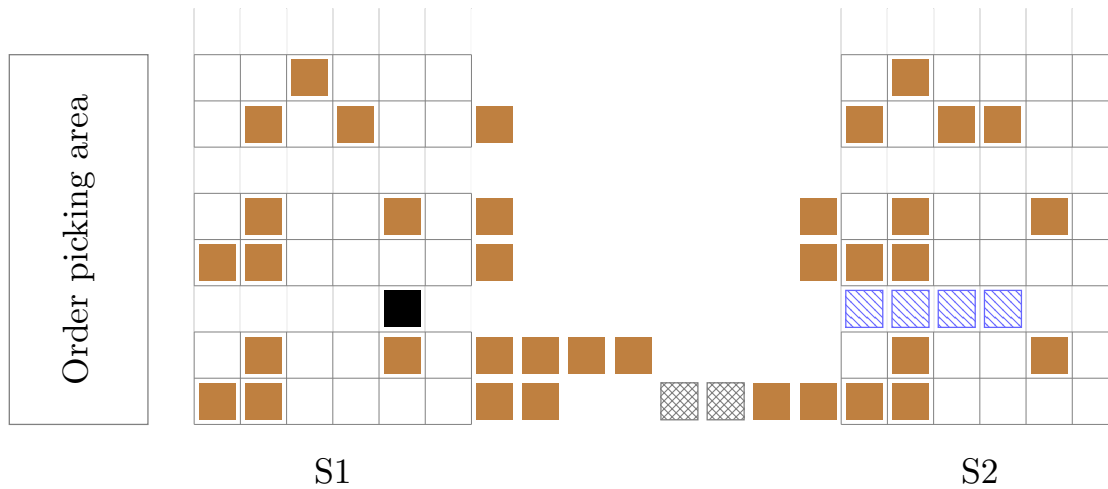


Figure 1.15: A schematic representation of the queues formed in the aisle between the storage rack sets S1 and S2. The black square indicates a high lift inside an aisle and the brown squares indicate pallets. The grey cross hatched squares represent open spaces in the inbound queue and the blue striped squared are pallets that are temporarily stored in a neighbouring aisle.

become too lengthy the space between opposite queues becomes so small that high lifts are unable to pass through it. This forces high lifts to either wait for the queues to be cleared or take a different route, both slowing down the process of changing aisles drastically.

Due to the continuous transfer of pallets to queues alternative queueing space is needed when the length of a queue exceeds four pallets. If a queue is full but more pallets are destined for the aisle a temporary queueing extension is created in an unused neighbouring aisle. As a high lift stores a pallet, from the storage queue, pallet jacks replenish the queue with pallets from the temporary storage queue, usually in a neighbouring aisle. Referring to Figure 1.15 the solid black block indicates a high lift storing pallets in S1. Four pallets are stored in the queue while the excess pallets, the blue striped squares, are stored in a temporary neighbouring aisle in S2 due to insufficient space. Regardless of a pallet's starting location in the DC it is always required to be placed in a queue before it is stored in the storage racks.

1.9 Scope and objectives

The management of PEP's Durban DC believes that a bottle neck is formed when pallets are moved through the storing racks. The bottleneck is caused by a high inflow rate into the storage rack system but a low outflow rate of pallets. It is believed that the low outflow rate is caused by the long travel distances and times to complete each pallet movement. The focus of this study is thus to find alternative methods of reducing the total travel time and distance when moving pallets through the storage racks. Alternatively high lift movement logics are investigated and analysed to determine the effect each has on the storage rack system. The effect of completing the jobs in a different sequence is also investigated. Lastly alternative slotting configurations are designed in an attempt to reduce the total travel distance and time. The proposed logics and algorithms are only applied to the storage rack set closest to the order picking area, S1. The operations conducted within storage rack set S1 and S2 are identical. This means that then analysis conducted on S1 can directly be applied to S2. These logics and algorithms only affect the movement of high lifts within the storage racks, ignoring the movements performed by pallet jacks and fork lifts.

A simulation model is coded to accurately replicate the real world scenario. Combinations of the proposed logics and algorithms are simulated using both a generated and historical job list. The results of each simulation is analysed to interpret the impact each configuration would have on the real world system. The proposed combinations are also compared with the current system employed by PEP.

The main objective of this study is to investigate PEP's Durban DC and find an improved model that PEP can employ regarding the movement of pallets in and out of the storage rack system by means of high lifts. To achieve this objective the problem is broken into sub objectives which can be summarised as follows:

1. Identify the problems associated with the current system that PEP uses.
2. Investigate alternative logics and algorithms to improve on the current system by reducing the completion time and distance for the same set of jobs.
3. Find accurate and sensible travel times for the three movement types.
4. Compile and analyse a job list from the historical data.
5. Formulate and build a simulation that models the current system.
6. Simulate the proposed movement logics and algorithms.
7. Interpret the results and make suggestions and recommendations.

1.10 Thesis layout

A detailed description of the activities in and around the storage rack system is given in Chapter 1. In Chapter 2 five alternative high lift movement logics are presented and discussed in detail. Five different heuristics are also investigated to find a good sequence to complete a job list. Lastly the effect of slotting pallets using different configurations are discussed.

Chapter 3 introduces the historical data received from PEP. An in-depth discussion is provided on the method used to clean the data and make it usable and the different anomalies found in the data. Further cleaned data is analysed and validated in order to determine accurate and sensible travel times for every process.

Chapter 4 shows the steps followed to validate and verify the simulation model. The logic behind the construction of the simulation model is discussed. The graphical user interface of the simulation is explained and shown to allow the reader to better understand what the simulation model is capable of.

The results and analysis obtained from the simulation model is presented and interpreted in Chapter 5. In the final chapter some concluding remarks about the simulation, the data and the different high lift logics are made. It also includes some topics for future studies.

CHAPTER 2

Alternative logics and algorithms

This chapter aims to find alternative movement logics and algorithms to improve the current system. Alternative movement logics are presented that attempt to decrease the dead-heading present when storing and retrieving pallets. Algorithms to find better sequences in which jobs are processed are investigated and finally the effect where pallets are stored is analysed.

2.1 High lift movement logics

A high lift's operating time is dependant on its travel distance and accumulated as pallets are processed. Streamlining the processes and reducing the travel time within the storage rack area will lead to reduced costs due to a through flow of pallets.

The action of a high lift moving without a pallet is known as dead-heading [2] and occurs between the time one pallet is dropped off and the next one is picked up. The travel times can be reduced by decreasing the travel distance. The best way of achieving this is by decreasing the dead-heading distance.

A single sequence of pallets, containing pallets to be stored and retrieved are used for analysis. This sequence of pallets is also referred to as a job list. A job list is defined as a collection of pallets that are stored, retrieved or moved in a sequence. A job refers to a single pallet in this job list that is stored, retrieved or moved by a high lift. Assuming that the job list contains I pallets that need to be stored and O pallets that need to be removed from storage. The total number of pallets being processed in the job list is then $K = O + I$.

Five high lift movement logics are develop to attempt to reduce the total movement distance and decrease the time to process all jobs in the job list. The five high lift movement logics that are developed are the **Single job with opposite side queueing (OS)**, **Single job with same side queueing (SS)**, **Paired jobs with same side queueing (SSP)**, **Paired jobs with opposite queueing (OSP)** and **Paired jobs with cyclic queueing (CS)**. Using this generally defined job list the movement logics are investigated. In Section A.1 an example of each of these movement logics is supplied to further explain each movement logic.

2.1.1 Single job with opposite side queueing (OS)

This movement logic is the logic currently followed by the high lifts in PEP's Durban DC. In the opposite side queueing system a storage movement is performed by the high lift picking up a pallet from the storage queue. The high lift travels with the pallet to the designated rack location and the pallet is stored. This step is illustrated in Figure 2.1 as the solid line. After storing the pallet the high lift dead-heads (moves without a pallet) back to the storage queue to pick up the next pallet. This step is illustrated with the right side dashed line in Figure 2.1. These two steps are repeated until the current batch of storage jobs are completed for the aisle. In the case of a different aisle having storage jobs the high lift would move to the aisle and proceed to complete the new batch of storage jobs. If no more pallets need to be stored the high lift travels empty to the rack location of the first retrieval job, illustrated by the left side dashed line in Figure 2.1. After a pallet is retrieved from the rack location the high lift moves it to the retrieval queue, illustrated by the zig zag line in Figure 2.1. The process is repeated until all retrieval jobs are removed from the storage rack system.

The total distance travelled by a high lift, to complete the job list using this movement logic, is calculated by summing the total distance a high lift travels carrying a pallet (D_P) and the distance a high lift is dead-heading (D_H) using equation (2.1) and (2.2). Here x_i represents the storage rack column and y_i represents the rack level of the rack location used for pallet i . The variable N denotes the total number of storage rack columns. Using this movement logic forces the high lift to dead-head a minimum distance equal to the distance travelled with a pallet. In the case of a high lift changing from storage jobs to retrieving jobs the dead-heading is the distance between the two pallet locations. The total distance travel could be calculated as

$$D_P = \sum_{i=1}^I [N - x_i + y_i] + \sum_{i=I+1}^K [x_i + y_i - 1] \quad (2.1)$$

and

$$D_H = \sum_{i=1}^{I-1} [N - x_i + y_i - 1] + \sum_{i=I+2}^K [x_i + y_i - 1], \quad (2.2)$$

$$+ |x_I - x_{I+1}| + |y_I - y_{I+1}|.$$

Focusing on a single pallet a high lift will travel the distance of two aisle lengths, one while carrying a pallet and one dead-heading, to move the pallet from the storage queue to the retrieval queue — irrespective of where it is stored in the storage rack system. This is due to the storing and retrieval queues being on opposite side of the storage rack.

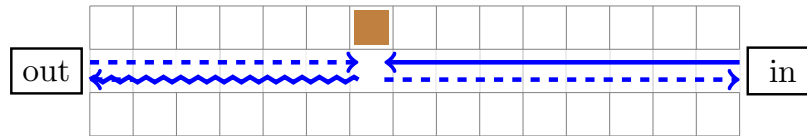


Figure 2.1: A schematic representation of the single job system with opposite end queueing used by PEP. The dashed arrow represents dead-heading, the solid arrow indicates a high lift storage movement and the jagged arrow indicates a retrieval movement. The dark square indicates the rack location of the stored pallet. The **in** and **out** blocks refer to the storage and retrieval queues respectively.

2.1.2 Single job with same side queueing (SS)

The system with queueing on the same side and doing a single job at a time is similar to the system discussed in § 2.1.1. A high lift picks up a pallet from the storage queue located closest to the order picking area and moves it to the designated rack location. This step is illustrated in Figure 2.2 by the solid line. The high lift then returns to the storage queue to collect the next pallet and repeats the process until all storage pallets are stored. If another aisle has pallets that needs to be stored the high lift moves to the next aisle and start storing pallets until no more aisles have storage jobs remaining. The high lift then starts removing pallets from the storage racks, illustrated by the zig zag line in Figure 2.2. After each storage and retrieval job the high lift dead-head the same distance as the completed job. This is illustrated in Figure 2.2 with the dashed lines. Using this movement logic the retrieval queue is situated on the same end as the storage queue as depicted in Figure 2.2.

The occupied distance (D_P) and dead-heading distance (D_H) are calculated as

$$D_P = \sum_{i=1}^K [x_i + y_i - 1] \quad (2.3)$$

and

$$D_H = \sum_{i=1}^{I-1} [x_i + y_i - 1] + \sum_{i=I+2}^K [x_i + y_i - 1] + |x_I - x_{I+1}| + |y_I - y_{I+1}|. \quad (2.4)$$

Here x_i represents the storage rack column and y_i represents the rack level of the rack location used for pallet i for each job respectively. The high lift will always travel the same distance to remove a pallet as it travelled to store it. In a scenario where pallets are placed uniformly in the entire storage rack the average total travel distance the high lift will travel is the same distance as with the movement logic described in § 2.1.1. Cases where pallets are stored and removed at a distance, less than half the length of the rack, from the queues this movement logic out performs the previous mentioned movement logic. If pallets are stored or removed from rack locations further than halfway into the storage rack this movement logic starts to under perform compared to the OS movement logic. Depending on the ratio of pallets stored/removed a distance less than half the lengths of the rack and pallets stored/removed a distance greater than half the rack this can either perform better or worse than the OS movement logic described in § 2.1.1.

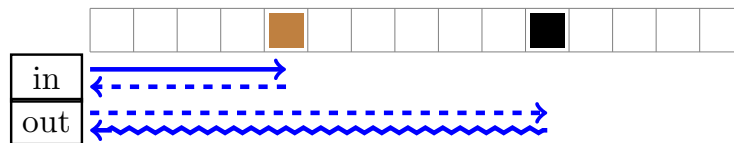


Figure 2.2: A schematic representation of a single job system with same side queueing. The solid arrow indicates a storage movement of the high lift with a load, the dashed arrow indicates the total dead-heading distance and the jagged arrow indicates a retrieval movement with a load. The **in** and **out** blocks refer to the storage and retrieval queues respectively.

2.1.3 Paired jobs with same side queuing (SSP)

A well known approach for reducing dead-heading is to pair a storage and retrieval job [2]. In this approach a storage and retrieval job is paired. As the first alternative to the single job method it is suggested that the storage queue and the retrieval queue both be on the same end as indicated in Figure 2.3 in an attempt to minimise the travel distance. The high lift starts at the storage queue moving a pallet to its designated storing location, illustrated by the solid line in Figure 2.3. After the pallet is stored the high lift directly moves to retrieve the paired retrieval pallet. Illustrated by the zig zag line in Figure 2.3 the retrieved pallet is moved to the retrieval queue. By moving directly from the rack location of the stored pallet to the rack location of the pallet that is retrieved there is only one instance of dead-heading. This is illustrated by the dashed line in Figure 2.3. An important property of the paired jobs system is that dead-heading and travel distance are directly dependant on the pairings and can be reduced by alternating the pairing of the storage and retrieval jobs.

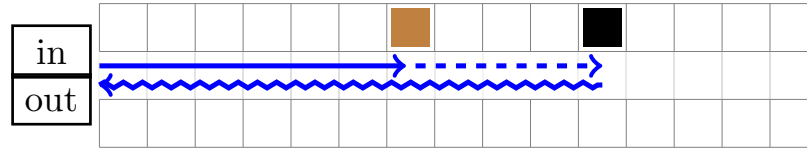


Figure 2.3: A schematic representation of a paired job system with same end queuing. The solid arrow indicates a storage movement of the high lift with a load, the dashed arrow indicates the total dead-heading distance and the jagged arrow indicates a retrieval movement with a load. The in and out blocks refer to the storage and retrieval queues respectively.

The occupied distance (D_P) and dead-heading distance (D_H) are calculated as

$$D_P = \sum_{i=1}^K [x_i + y_i - 1] \quad (2.5)$$

and

$$D_H = \sum_{i=0}^{\frac{K}{2}-1} [|x_{2i+1} - x_{2i+2}| + |y_{2i+1} - y_{2i+2}|]. \quad (2.6)$$

Here x_i represents the storage rack column and y_i represents the rack level of the rack location used for pallet i for each job respectively. Dead-heading only occurs when the high lift moves from the rack location where a pallet was stored to the locations where the paired pallet is retrieved. This means that with a job list with equal storage and retrieval jobs dead-heading only occurs once per job pairing.

A practical problem that could arise when queuing at the same end of the aisles is that there would be a large increase in traffic in the area between the storage racks and the order picking area. To accommodate the increase of pallets in the queueing area at the end of the storage rack system more space needs to be made available. This can be done by moving the rack locations that are currently at the front to the back, opening more space in the front for movement.

This could, however, be expensive or difficult to incorporate. An alternative is to introduce queues that are underneath the storage rack with pallets rolling on rollers for ease of access. In an attempt to counter the increase of pallets in one side of the storage rack system the next proposal is given.

2.1.4 Paired jobs with opposite side queuing (OSP)

The next method that is considered is a combination of the systems discussed in § 2.1.1 and § 2.1.3. It is a paired job system, but with the storage and retrieval queues on opposite ends of the aisle, as illustrated in Figure 2.4. A pallet is picked up by the high lift from a storage queue and moved to the designated rack location. This is illustrated by the solid line in Figure 2.4. After the pallet is stored the high lift dead-heads to the location of the retrieval pallet and removes it from the storage rack. Depicted as the zig zag line in Figure 2.4 the pallet is moved to the retrieval queue. After the pallet is placed in the retrieval queue the high lift dead-head back to the storage queue through the aisle. The dead-heading is illustrated in Figure 2.4 with the dashed lines. This repeats until all jobs have been stored and retrieved.

The occupied distance (D_P) and dead-heading distance (D_H) may be calculated as

$$D_P = \sum_{i=0}^{\frac{K}{2}-1} [(N - x_{2i+1} + y_{2i+1}) + (x_{2i+2} + y_{2i+2} - 1)] \quad (2.7)$$

and

$$D_H = \sum_{i=0}^{\frac{K}{2}-1} [|x_{2i+1} - x_{2i+2}| + |y_{2i+1} - y_{2i+2}| + (N + 1)]. \quad (2.8)$$

Once again x_i represents the storage rack column and y_i represents the rack level of the rack location used for pallet i for each job respectively. Because storage and retrieval jobs are paired, it is required to only do each part of equation (2.7) and (2.8) once per job pairing. Equation (2.7) and (2.8) show that when using this movement logic there will always be a minimum of one aisle's length of dead-heading per job pairing. Added to this is the dead-heading between the rack locations of the storage and retrieval pallet.

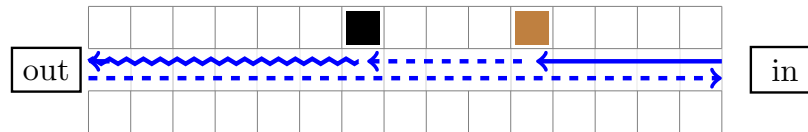


Figure 2.4: A schematic representation of the movement of high lift processing two paired jobs with an opposite end queuing system. The solid arrow represents a storage movement, the jagged arrow indicates a retrieval movement and the dashed arrow indicates dead-heading. The in and out blocks refer to the storage and retrieval queues respectively.

2.1.5 Paired jobs with cyclic queueing (CS)

In a further attempt to decrease the total travel distance and time the paired jobs with cyclic queueing movement logic is introduced. This approach is a paired job system that allows the high lift to work in two directions. It makes use of a storage and retrieval queue on both ends of the aisle. This layout is illustrated in Figure 2.5 where there are four queues for each aisle. This approach is similar to the OSP system discussed in § 2.1.4 where the storage and retrieval queues for each paired job are on opposite ends of the aisle. In this logic the high lift, however, does not return to the storage queue from where it originated from. Rather, the storage pallet of the new paired jobs is removed from the storage queue situated at the same end of the aisle the high lift is at currently. The process of storing a pallet is illustrated in Figure 2.5 as solid lines and the process of retrieving a pallet as the zig zag lines. The method is repeated, switching the end at which the storage and retrieval queues are after each job pairing is completed, until no more jobs need to be completed.

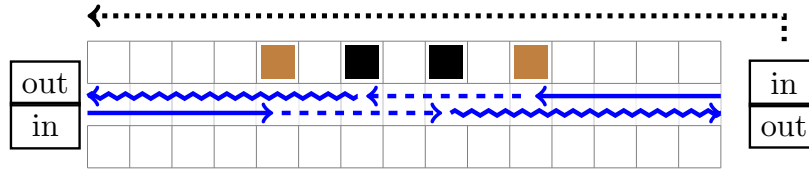


Figure 2.5: A schematic representation of the movement of high lifts processing four jobs with a paired cyclic system. The solid arrow represents a storage movement, the jagged arrow indicates a retrieval movement and the dashed arrow indicates dead-heading. The dotted arrow shows the shortest route for a pallet jack to move from the furthest outbound queue to the picking lines. The in and out blocks refer to the storage and retrieval queues respectively.

The distance a high lift moves with a pallet is split into two separate equations D_{P1} , equation (2.9), and D_{P2} , equation (2.10), depending on the direction the high lift is moving. Variable D_{P1} calculates the movement distance to complete a job pairing when the storage queue is situated at the same end of the aisle than the order picking area. Variable D_{P2} represents the opposite with the storage queue being at the end of the aisle between rack sets S1 and S2. The distance can be calculated as

$$D_{P1} = \sum_{i=0}^{\frac{K}{2}-4} [N - x_{4i+1} + y_{4i+1} + x_{4i+2} + y_{4i+2} - 1], \quad (2.9)$$

$$D_{P2} = \sum_{i=1}^{\lfloor \frac{K}{4} \rfloor} [N + x_{4i+3} + y_{4i+3} - x_{4i+4} + y_{4i+4} - 1] \quad (2.10)$$

and

$$D_H = \sum_{i=0}^{\frac{K}{2}-1} |x_{2i+1} - x_{2i+2}| + |y_{2i+1} - y_{2i+2}|. \quad (2.11)$$

This logic relies on the utilization of the manual labour, in the form of pallet jacks, as approximately half of the retrieved pallets are placed on the far-end of the aisle. This means pallet jacks will be required to transport pallets to the order picking area and will always try to move pallets using the shortest route. This route is illustrated in Figure 2.5 as the dotted line. Due to the queues being situated between storage rack sets S1 and S2 the shortest route would be through an aisle. Pallet jacks that move through the aisles increase the risk of injury, by accidentally moving in front of a high lift, as well as occupying aisles while they are in them, making it unavailable for high lifts to enter.

2.2 Job pairing sequence

In an attempt to further reduce the movement distance, the effect of changing the sequence in which jobs are completed and paired are investigated. Finding the best sequence for the job list is not a simple task, especially when the list consists of a large number of jobs.

Before a good sequence for a job list can be found, an initial job list is generated. The job list can either be generated using historical data or by means of a script. Depending on the type of high lift movement logic used the job list is generated differently. In the case where a job pairing movement logic is used the storage jobs are paired with retrieval jobs until no more pairings can be formed, this happens when there is not an equal number of storage and retrieval jobs. When a movement logic is used that does not make use of job pairing all storage jobs are moved to the top of the job list. Two methods are implemented in an attempt to find good job pairings, namely a Tabu Search meta-heuristic [9] and the Hungarian algorithm [16].

Utilising the Tabu Search meta-heuristic the optimal job sequence is reached by swapping jobs. A swap is accepted if it decreases the total distance to complete the job list. In the case where no swaps decrease the total distance, jobs are swapped that do not affect the total distance. These swaps that do not affect the total movement distance are done in an attempt to escape local minimums. The candidates for swapping are defined as the jobs with the most dominant movement type. This means that in a job list if there are more storage jobs than retrieval jobs the storage jobs would be used for swap. The search neighbourhood are all jobs considered to be swapped with. Finally a tabu list is created for swaps that are accepted in an attempt to avoid cycles forming. Cycles form when the same jobs are repeatedly swapped but never improve the solution. By adding jobs that have successfully been swapped to the tabu list, hopefully reduces the chances of a cycle forming.

To find a good sequence of a job list four Tabu Search algorithms are considered. The four tabu search algorithms are **Random pairing**, **Pairing from the top**, **Pairing from the best** and **Pairing from the worst**.

Algorithm (1) is an extract that is present in all four tabu search algorithms. In this extract it is assumed the best solution and tabu list have been initialised and candidates have been found. In line 1 a temporary solution is obtained by swapping two candidates in the solution. The total distance to complete the new job list is compared to that of the current best solutions (line 2). If the temporary solution is better it becomes the new best solution and the candidate list is added to the tabu list (line 3–4). The candidates are added to the tabu list instead of the solution to reduce the memory stored in the tabu list. Lines 5–8 maintain the length of the tabu list if the length of the tabu list exceeds the user-defined length, tabu entries are removed. In these algorithms, the tabu entries that have been in the tabu list the longest are removed first.

Algorithm 1: General Tabu search extract: testFitness(sbest,candidateList,tabuList)

Input : Current best solution, Candidate list and tabu list**Output:** new best solution.

```

1 sCandidate ← Exchange(sBest,candidateList);
2 if (fitness(sCandidate) < fitness(sBest)) then
3   sBest ← sCandidate;
4   tabuList ← addElement(candidateList);
5   while (size(tabuList) > maxTabuListSize) do
6     removeElement(tabuList)
7   end
8 end

```

Algorithm 2: Random Tabu Search

Input : List of jobs in any sequence and movement type**Output:** Good sequenced list of jobs.

```

1 s ← s0;
2 sBest ← s;
3 tabuList ← null;
4 while (not stopppingCondition()) do
5   candidateList ← null;
6   while (candidateList ← null) do
7     Candidate1 ← getRandom();
8     Candidate2 ← getRandom();
9     if (Candidate1 ≠ Candidate2 and mostCommon(Candidate1) and
        mostCommon(Candidate2)) then
10      Candidate ← Candidate + Candidate1 + Candidate2;
11      if (not containsTabuElements(Candidate,tabuList)) then
12        candidateList ← candidateList + Candidate;
13      end
14    end
15  end
16  testFitness(sbest,candidateList,tabuList)
17 end

```

Random pairing — In Algorithm (2) two jobs with the same movement type are selected at random to be swapped. Lines 1–3 represent the initial setup of the algorithm respectively defining the initial solution as an imported job list, setting the initial solution as the current best observed solution and initialising an empty tabu list.

From line 4 the actual algorithm starts. As long as the user-defined stopping criteria, in this case a number of iterations, is not met, the algorithm continues to try and improve the sequence. In line 5 a candidate list is initialised that will temporally store jobs than can be swapped. The loop in line 6 repeats until two candidates are selected. Two jobs are randomly selected from the entire job list (line 7–8). Only if the randomly selected jobs are not the same job and both their movement types are most dominant are they considered as candidates (line 9). The two selected jobs are placed in a short term candidate list (line 10). This candidate list's only purpose is to be used to confirm whether or not the action of swapping the two candidates have been performed before (line 11). If the two candidates have not been swapped yet, according to

the tabu list (line 11), they are added to the candidate list (line 12). The current- best solution, candidate list and tabu list are used in Algorithm (1) to determine if swapping the candidates results in a better solution (line 16).

Pairing from the top — Algorithm 3 attempts to improve the initial job list by staring at the top of the job list and swapping jobs with the same movement type consecutively. The initial setup of the algorithm is performed, with the initial solution being the imported job list, the current best observed solution set as the initial solution and an empty tabu list being initialised.

The tabu search starts in line 4. The algorithm continuously searches for candidates to reduce the distance until an user-defined stopping criteria is met. Here it is a fixed number of iterations. Starting at the top of the job list each job is inspected (line 5). If the job's movement type is most dominant it is selected as a candidate (line 6–7). With the first candidate selected the algorithm again steps through the entire job list starting at the top (line 8). Each job is considered a candidate (line 10) but is only considered if it is not the same job as the other candidate and it is the most dominant movement type (line 11). The candidates are placed in a short term candidate list (line 12). If the short term candidate list is not in the tabu list the candidates are selected to be swapped (line 10–12). The current- best solution, candidate list and tabu list are sent to Algorithm 1 to determine if swapping the candidates results in a better solution (line 17).

Algorithm 3: Pairing from top Tabu Search

Input : List of jobs in any sequence and movement type

Output: Good sequenced list of jobs.

```

1   $s \leftarrow s_0$ ;
2   $s_{\text{Best}} \leftarrow s$ ;
3   $\text{tabuList} \leftarrow \text{null}$ ;
4  while (not stoppingCondition()) do
5      for ( $i < \text{length}(s)$ ) do
6          if mostCommon( $s_i$ ) then
7               $\text{Candidate1} \leftarrow s_i$ ;
8              for ( $j < \text{length}(s)$ ) do
9                   $\text{candidateList} \leftarrow \text{null}$ ;
10                  $\text{Candidate2} \leftarrow s_j$ ;
11                 if ( $\text{Candidate1} \neq \text{Candidate2}$  and mostCommon( $\text{Candidate2}$ )) then
12                      $\text{Candidate} \leftarrow \text{Candidate} + \text{Candidate1} + \text{Candidate2}$ ;
13                     if (not containsTabuElements( $\text{Candidate}$ ,  $\text{tabuList}$ )) then
14                          $\text{candidateList} \leftarrow \text{candidateList} + \text{Candidate}$ ;
15                     end
16                 end
17                 testFitness( $s_{\text{best}}$ ,  $\text{candidateList}$ ,  $\text{tabuList}$ )
18             end
19         end
20     end
21 end

```

Pairing from the best — Algorithm 4 starts by finding the job pairing that contributes the least to the total distance. In this algorithm there are, however, two tabu lists – one for the swaps and one for the best pairings.

Lines 1–2 represent the initial setup of Algorithm 4. The initial solution is an imported job list

and the initial solution is set as the current best observed solution. The distance of the current best paired jobs is initialised with a value of infinity (line 3). Lines 4–5 initialises the tabu lists for swaps and best pairing respectively (lines 4–5). The purpose of the best pairing tabu list is to ensure the algorithm does not select the same best pairing each iteration — causing the solution space to stagnate.

The tabu search starts in line 6. Until the user-defined stopping criteria is met the algorithm continuously attempts to find a better solution. Lines 7–25 attempts to find the job pairing that influences the total distance the least. Starting at the top of the job list, two jobs are selected as a possible pair (line 7–9). Due to the manner in which possible pairings are selected, the current and next job, the algorithm performs one less iteration than there are jobs. Only if the two selected jobs have different movement types are they added to the short term candidate list (line 10–12). In line 13 it is determined if this pairing was selected as the best before by comparing it to the entries in the pairing tabu list. Only if the influence of the paired candidate jobs is less than the current best pairing’s distance are the jobs accepted as the new best pairing (line 14–16). The new best job pairing is then added to the paired tabu list (line 18). If the paired tabu list exceeds the user-defined lengths entries are removed (line 19–20) — starting with the oldest entries.

Lines 26–36 use the best job pairing and attempts to find a candidate that can reduce the pairing influence even further. Line 26 determines which job of the pair is going to be used as a candidate to reduce the distance of the entire job list based on the most dominant movement type. Iterating through the job list, from the top, every job is considered (line 27–30). A job is only accepted as a candidate if it is not the same job as the other candidate and both candidates have the same movement type (line 31). If these two candidates have not yet been swapped according to the tabu list (line 33) they are sent as candidates to Algorithm 1 along with the current- best solution and tabu list.

Pairing from the worst — This algorithm starts by finding the job pairing that affects the total distance the most. This algorithm is the exact opposite of Algorithm (4) with only two small changes occurring, see Algorithm 5 in Appendix A.2.

Once the worst job pairing is found the job with the most dominant movement type is chosen as the candidate. This candidate is temporarily swapped with all other jobs of the same movement type. Each swap’s effect on the total distance is calculated and the swap that decreases the total distance the most is accepted. The accepted swap is added to the tabu list. The process is repeated until the stopping criteria is reached accepting only swaps that are not in the tabu list and decreases the total distance.

Hungarian method — When using the Hungarian method the sequencing problem is handled as an assignment problem. A matrix is constructed where each row represents the dominant job type and the columns represent the remaining job type. Each entry into the matrix is the movement distance required to complete the job pairing. In the case that there is an unequal number of job types dummy job place holders are added. These dummy jobs represent the distance required if a job is completed without a pairing.

2.3 Pallet slotting problem

Pallet slotting is the process of determining the most convenient positions within the storage rack system to place pallets in order to reduce labour. It is believed that a well-slotted DC will always outperform one that applied limited effort to improve on slotting [24].

The two most commonly used types of slotting are random slotting and fixed slotting [24].

Random slotting implies that pallets are not assigned to specific storing locations but to any random available location. Fixed slotting implies that pallets are assigned to fixed storing locations based on specific characteristics of the pallet [24].

Algorithm 4: Best pairing Tabu search

Input : List of jobs in any sequence and movement type

Output: Good sequenced list of jobs.

```

1   $s \leftarrow s_0$ ;
2   $sBest \leftarrow s$ ;
3   $pBest \leftarrow \infty$ ;
4   $tabuList \leftarrow \text{null}$ ;
5   $bestTabuList \leftarrow \text{null}$ ;
6  while (not stoppingCondition()) do
7      for  $i < \text{length}(s)-1$  do
8           $pCandidate1 \leftarrow s_i$ ;
9           $pCandidate2 \leftarrow s_{i+1}$ ;
10          $pCandidate \leftarrow \text{null}$ ;
11         if  $\text{jobType}(pCandidate1) = \text{Storage}$  and  $\text{jobType}(pCandidate2) = \text{Retrieval}$  then
12              $pCandidate \leftarrow pCandidate + pCandidate1 + pCandidate2$ ;
13             if (not containsTabuElements( $pCandidate, bestTabuList$ )) then
14                 if  $\text{fitness}(\text{tempSequence}(pCandidate1, pCandidate2)) < bBest$  then
15                      $Candidate1 \leftarrow pCandidate1$ ;
16                      $Candidate2 \leftarrow pCandidate2$ ;
17                      $bBest \leftarrow \text{fitness}(\text{tempSequence}(Candidate1, Candidate2))$ ;
18                      $bestTabuList \leftarrow \text{addElement}(pCandidate)$ ;
19                     while ( $\text{size}(bestTabuList) > \text{maxTabuListSize}$ ) do
20                          $\text{removeElement}(bestTabuList)$ 
21                     end
22                 end
23             end
24         end
25     end
26      $Candidate1 \leftarrow \text{mostFrequent}(Candidate1, Candidate2)$ ;
27     for ( $j < \text{length}(s)$ ) do
28          $candidateList \leftarrow \text{null}$ ;
29          $Candidate \leftarrow \text{null}$ ;
30          $Candidate2 \leftarrow s_j$ ;
31         if ( $Candidate1 \neq Candidate2$  and  $\text{mostCommon}(Candidate2)$ ) then
32              $Candidate \leftarrow Candidate + Candidate1 + Candidate2$ ;
33             if (not containsTabuElements( $Candidate, tabuList$ )) then
34                  $candidateList \leftarrow candidateList + Candidate$ ;
35             end
36         end
37          $\text{testFitness}(sbest, candidateList, tabuList)$ 
38     end
39 end

```

It is agreed in industry that the fastest moving pallets should be stored in the most convenient

locations. The most convenient locations are defined as the storing locations closest to the receiving and shipping areas [2]. Before pallets can be slotted their classification should be determined.

2.3.1 Pallet classification

There are various ways in which to classify pallets. Winston [44] mentions one method known as ABC classification. In this method pallets are ranked based on their monetary value. Bartholdi [2] extends this method by stating that pallets can also be classified according to how many times they need to be handled. Pallets with high rankings are seen as favourable and will enjoy prime storage locations as seen in Figure 2.6.

Another method for the classification of pallets can be based on the length of their replenishment cycle (RC). Some stock needs to be replenished often whereas other stock might not. Frequently replenished stock is referred to as fast moving stock. This makes it possible to classify pallets according to the length of the replenishment cycle, for instance stock that typically enter and leave a DC in less than a week is considered RC1.

RC value storing is a strategy for allocating storage locations in a storage rack to pallets in an attempt to streamline the movement within the pallet rack system. It assigns a location to each pallet based on the length of its RC. Fast moving stock usually have more strict deadlines and therefore can be retrieved quicker if they are grouped together close to their destination. Therefore one method, as mentioned by Bartholdi [2], suggests that the most convenient locations are reserved for fast moving pallets close to the picking lines as depicted in Figure 2.6.

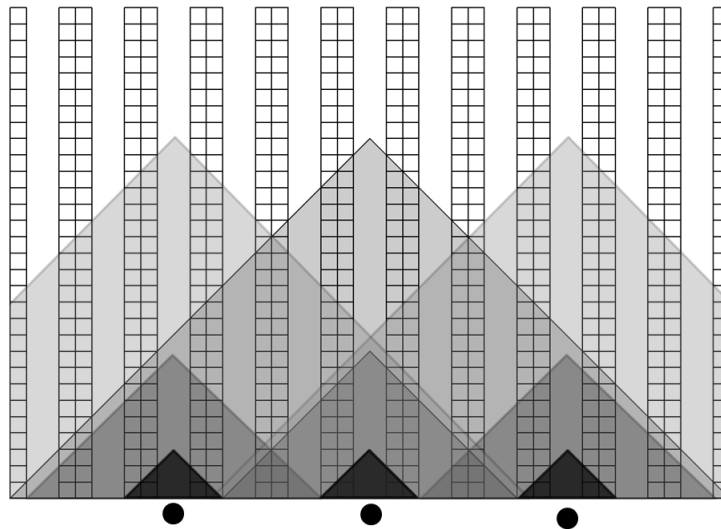


Figure 2.6: A schematic representation of where pallets should be stored based on their RC values. The circles represent outbound destinations, like picking lines. The different shades indicate where different RC valued pallets should be stored. Darker shaded areas indicate where the fastest moving pallets should be stored and slow moving pallets should be stored in the lighter shaded areas [2].

2.3.2 Replenishment cycle pallet slotting

For this paper five different variations of RC value storing implementations are investigated. Figure 2.7(a) shows the first variation where the storage rack is divided into columns based on the RC value. Pallets with any RC value can be stored close to the ground making this configuration ideal for DC's where vertical movement is more expensive for a high lift. This configuration prefers RC value distributions with lower RC values due to higher RC values being stored very far from the retrieval queue.

The second variation, Figure 2.7(b), is the inverse of the first. In this variation the storage racks are being divided into rows based on the RC value. Using this configuration pallets are stored higher in the storage rack. This makes it possible for pallets with any RC value to be stored relatively close to the retrieval queue. This configuration is preferred by DC's where it is expensive for high lifts to move in the horizontal direction. The row configuration prefers a uniform distribution of RC values. The more abundant a specific RC value is the further the storing distance from the retrieval queue will become. The first two variations are straightforward being that they are constructed purely from a geometric view point. Figure 2.7(c) depicts a stepping configuration which is a hybrid consisting of the first two variations and the travel distance from the removal queue. This configuration allows pallets to be stored closer to the ground than the row configuration. At the same time it allows high RC valued pallets to be stored closer to the retrieval queue. This configuration can thus be used by a system where horizontal or vertical movements are expensive. This variation adds a layer of complexity when deciding at which distances specific RC values are allocated. The form of the stepping configuration is very important as making small changes can affect the movement distance. The last two variations are in essence the same with the only difference being for the one the RC allocation is determined by summing the horizontal and vertical distances, see Figure 2.7(d). The other variation uses Pythagoras [39] to calculate the distance from the retrieval queue. Determining the size or number of storage locations of each RC value for any type of DC is nearly impossible and should be determined for each DC independently. The best way to achieve this would be to use historical data.

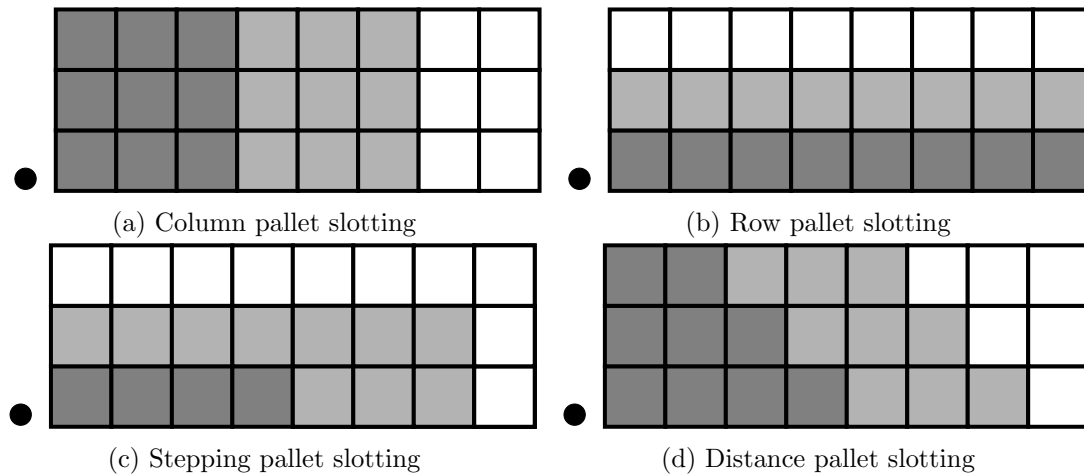


Figure 2.7: A schematic representation of the cross sections of two pallet racks with different RC value sections. The circles indicate outbound queues. The darkest shaded area represents storage locations for fast moving pallets, the lighter shaded area indicates storage location for slower moving pallets and the lightest shaded areas indicate the storage area for the slowest moving pallets, i.e. pallets that will be in storage for long periods of time.

All five these RC value storing variations are confronted with the possibility that there may be

a shortage of storage space for certain RC values if there is a high inflow rate of pallets with the same RC value. A possible solution would be to store the excess pallets in a different aisle with storage space for the pallets. Although this is a reasonable approach it would lead to a large increase in movement distance. Dedicating areas to specific RC values in the storage rack system make the storage and removal of pallets easier and more convenient for high lift operators who memorised the layout. Dedicated storage does not ensure that space is used efficiently with certain RC areas locations being empty while others have an overflow [2]. To reduce the risk of inefficiently storing pallets a strategy of using shared storage space should be used.

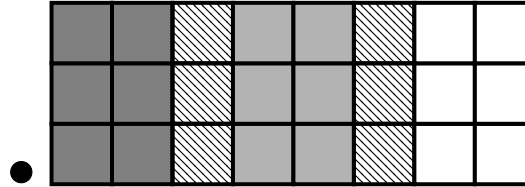


Figure 2.8: A schematic representation of how shared storage areas are added to the column pallet slotting configuration.

Using the RC value slotting variation from Figure 2.7(a) shared storage areas are added to the configuration. Figure 2.8 shows the newly added shared storage areas that allows pallets that do not have space in their designated RC storage area to be stored within the aisle. Although adding the shared storage areas ensures that the storage space is utilised more efficiently the idea of placing the fastest moving pallets closest to the retrieval queue should not be undermined by placing pallets with high RC values in shared storage close to the retrieval queue. Assigning a hierarchy to each shared storage area based on the RC values areas next to it allows pallets to be stored more efficiently. In the case that a dedicated RC area is over flowing the pallets are stored in the shared storage area next to the current RC area closest to the retrieval queue. If a shared storage area is full the next shared storage area, moving towards the end of the aisle, is inspected for available storage locations. Only after a pallet could not be stored would it be placed in a different aisle. For a DC with four different RC values (1, 2, 3, 4) there will be three shared storage areas (R1, R2, R3). When the RC1 and RC2 areas are filled these pallets are stored in R1 until this shared area is full. If R1 is filled the RC1 and RC2 pallets are placed in the R2 area until it is full after which these pallets are stored in R3. RC3 pallets are stored in R2 until it is filled after which it is stored in R3. When R3 is also full RC3 pallets are stored in R1. RC4 pallets are stored in R3 moving to R2 and then R1 as the shared areas become full.

CHAPTER 3

Data

Data capturing is the process of collecting information and structuring it to be readable and usable, often by computers [21]. When simulating real world scenarios the accuracy is greatly dependant on the data input used [28]. Data must be factual and relevant for a simulation to be meaningful and credible.

In today's business environment information is important and the approach used for collection, processing and distributing information is vital. The method used for gathering information can determine a company's productivity, product quality and overall competitiveness [42].

It is crucial that all captured data be timely, accurate and unbiased. Data is only as valuable as how accurately it was collected. When using people to collect data a number of things can go wrong and create problems with the data capturing process [29]. It is thus essential for companies to incorporate software that is capable of managing and recording data rapidly and accurately. This would ensure that resources are managed properly throughout the entire company [42].

3.1 Warehouse management software

Warehouse management software (WMS) is used to assist with the operations of a warehouse. WMS allows the management of a warehouse to be centralised making inventory control and the movement and storing of stock easier. It is possible for WMS to be the core application managing a warehouse on its own or be integrated into a larger system.

In most cases WMS data is automatically collected using radio-frequency portable terminals with bar code scanners [32]. Other less common data collection methods used by WMS are voice systems, Radio-frequency identification (RFID) or light-directed systems.

The data used in this study is collected using PEP's Durban DC WMS. When collecting data in the storage racks, bar code scanners are primarily used. Every worker who moves a pallet or individual box into or from the storage racks is equipped with a bar code scanner. The scanner has a unique ID to properly identify and track which worker performed a scan. Bar codes are attached to every individual box on a pallet. Each pallet also receives a bar code representing the grouping of boxes stored on it. Every storage cell has a bar code assigned that includes a unique location code. The bar codes allow the WMS to track stock and at the same time monitor employee efficiency. When a pallet is moved the location where it stood is scanned as well as the pallet. This allows the WMS to record all information regarding the location, time,

movement type and employee information. When the pallet is placed in its new location the pallet and location is scanned to ensure all information is recorded by the WMS.

3.2 Raw data

Before any of the raw data, received from PEP's IT department [38], can be used in the model it must first be validated. The provided data is stored in an Excel data sheet and consists of information regarding the movement of pallets. The data represents all the movements recorded in the storage rack area during the time interval 23-Feb-2015 to 23-Feb-2016. The file consists of 1 275 916 data points (jobs) in total and represents the movement of pallets that moved in the storage rack area. The dataset was constructed using the information gathered with the DC's WMS. The WMS is used in the DC to, amongst other things, record information regarding the movement of pallets in the storage racks. This information includes pallet characteristics, locations of pallets as well as the time at which workers scanned a pallet. Table 3.1 shows the format of the dataset that is received from PEP.

MOVE_ID	PALLET_ID	MOVE_TYPE	EMP_ID	START_DATETIME	END_DATETIME	ORIGIN_LOC	DEST_LOC	SKU_NO	REPL_CYCLE
723980	2479830	Linebuild	1961	04-Mar-2015 08:14:19	04-Mar-2015 08:16:53	LBD006	S131	198958	-
716901	2606776	Storage			27-Feb-2015 08:41:48	PSG005	AI05007	661722	14
717265	2590263	Linebuild	1627	02-Mar-2015 14:47:39	02-Mar-2015 14:48:37	PND415	K119	196073	-
716901	2606776	Storage	1617	27-Feb-2015 08:35:01	27-Feb-2015 08:37:46	MAR112	AI05007	661722	14
736499	2614920	Storage	1944	05-Mar-2015 22:14:54	05-Mar-2015 22:15:57	PND322	AV04025	199370	-
719059	2605017	Movement	1009	27-Feb-2015 12:00:29	27-Feb-2015 12:00:42	PND612	DH01003	129008	-
721132	2603153	Linebuild	1661	28-Feb-2015 14:39:17	28-Feb-2015 14:39:57		FF129	206214	-
725716	2610935	Storage	2004	03-Mar-2015 13:17:44	03-Mar-2015 13:18:22	PSG004	AG06019	200029	-
714619	2603640	Linebuild	1961	28-Feb-2015 13:46:31	28-Feb-2015 13:46:34	PND908	FF114	197620	-

Table 3.1: Table representing the format of the data received from PEP.

Each recorded data point contains the following information fields:

- **MOVE_ID** – An identification value assigned to represent the movement of a pallet.
- **PALLET_ID** – A unique value assigned to pallets allowing each individual pallet to be tracked effortlessly throughout the entire storage rack area.
- **MOVE_TYPE** – The type of action that is performed on the pallet in the storage rack. The three moving types considered in the data are as follows:

Storage – Indicates that a pallet is moved from a storage queue at the end of the aisle to the specified rack location in the storage racks system.

Linebuild – Indicates that a pallet is retrieved from a rack location in the storage racks system and placed in the retrieval queue.

Movement – Indicates that a pallet is relocated from its current rack location to a different rack location in the storage racks.

- **EMP_ID** – A unique value assigned to the worker (pallet jack, forklift or high lift operator) that performed the task of moving a pallet.
- **START_DATETIME** – The time at which the pallet was scanned to start the job (picked up).
- **END_DATETIME** – The time at which the pallet was scanned to finish the job (placed down).
- **ORIGIN_LOC** – A code representing the original location where the pallet is picked up from when the job starts.

- **DEST_LOC** – A code representing the destination location where the pallet was placed when the job is finished.
- **SKU_NO** – This number is assigned according to the type of stock keeping unit (SKU) stored on the pallet.
- **REPL_CYCLE** – The replenishment cycle showing the number of days the pallet is stored in the storage racks system before it is removed.

Depending on the location of a pallet its **ORIGIN_LOC** and **DEST_LOC** codes differ. The most important codes of all these are the seven character rack location codes e.g. **AA05025**. Considering the example **AA05025** the first two characters refer to the name of the aisle in which the rack location is situated. These character range from **AA** to **CM** representing all the aisles in the storage rack. The forth character indicates the height of the rack location in the specified aisle. The last two characters indicate the depth of a rack location into the aisle. The rack locations are numbered from a depth of 1 to 32, where one represents a depth of one rack location into the aisle from the end of the aisle closest to the order picking area.

Additionally the PND queueing code is important for understanding the data. Although it is not important to know the exact code it is important to know which pallets have this code assigned to them. PND is an abbreviation for “Pick and Drop” and refers to the storage and retrieval queue locations at the ends of the aisles. The code number can be ignored because pallets are always stored or retrieved from queues that are situated at the end of an aisle. All the other codes represent queues and storage areas outside the scope of this study.

3.3 Data validation

Although the data was recorded using a WMS the system is not entirely automated. In PEP’s DC it is still each workers responsibility to scan the pallets and their locations as job are being processed. This results in some human errors being made when recording data. Table 3.1 shows an example of human error where the empty spaces indicate movement actions not being recorded. All these errors should thus be addressed before the data can be used for analysis of the DC or in a simulation. Before any adjustments are made to the data set PEP is consulted to ensure no data points are removed that alters the credibility of the data set. The clean up of the data set comprises of three clean up phases. The first phase condenses the data set to the scope of this study. The second phase removes any anomalies observed in the data set. The third phase alters the travel times of the data points to accurately represent the observed times.

Phase 1 is split into three steps. Step 1 starts with removing all data points that have empty entries in **START_DATETIME**, **ORIGIN_LOC** or **DEST_LOC**. The analysis of the data is dependant on these fields and making any assumptions on the missing information could effect the analysis. Table 3.2 shows that (90 243) 7.07% of the data is removed due to empty entries.

Step	Error Type	Data points	Removed	Remove percentage	Remaining data
1	Empty fields	1 275 916	90 243	7.07%	1 185 673
2	Not in Racks	1 185 673	610 854	51.52%	574 819
3	Not part of scope	574 819	103 350	17.98%	471 469

Table 3.2: Table representing the number of data points removed from the data set during the phase 1 clean up process.

The second step is to remove all the recorded movements that were performed outside of the storage racks area. These movements were included in the original data set because the pallet on which the movement was performed moved through the storage racks area. However, these data points do not form part of the scope. These data points comprise of (610 854) 51.52% of the remaining data as shown in Table 3.2 and is removed.

Finally, any job that was not completed in storage rack set S1 is removed. This means that only data representing pallets stored or retrieved from aisles AA – BT is considered when analysing the data set. Table 3.2 shows that (103 350) 17.98% of the remaining data points are removed. The total number of data points are reduced by (804 447) 63.05% to 471 469 when removing the all the data points that fall outside of the scope.

Although the WMS was implemented on 16 June 2015 there are still a number of issues regarding the data. In phase 2 of the clean up process the data set is inspected further and four prominent anomalies are discovered as listed in Table 3.3.

Anomaly	PALLET_ID	MOVE_TYPE	START_TIME	END_TIME	ORIG_LOC	DEST_LOC
1	3518415	Movement	11:17:30	11:17:35	AG03013	AG03013
	3515268	Movement	16:04:23	16:04:29	BR01001	BR01001
	3281147	Movement	04:53:35	04:57:22	AO02014	AO02014
2	3519723	Storage	15:49:44	15:50:51	MAB118	AP03001
	3519723	Storage	15:49:44	15:50:51	MAB118	AP03001
	3519723	Storage	16:12:52	16:13:11	PND215	AP03001
	3519723	Storage	16:12:52	16:13:11	PND215	AP03001
3a	3519031	Movement	23:50:49	23:50:52	LBD011	AY01013
	3519032	Movement	23:50:55	23:50:57	LBD011	AY01013
	3519018	Movement	23:51:00	23:51:02	LBD010	AY01013
3b	3318647	Linebuild	20:48:59	20:49:04	BG01028	J121
	3318644	Linebuild	20:49:19	20:49:26	BG01028	J119
	3318646	Linebuild	20:49:48	20:49:55	BG01028	J123
4	1568379	Linebuild	03:54:30	03:55:30	BL06005	O114
	1568379	Linebuild	03:54:30	03:55:30	BL06005	O114
	1568379	Linebuild	03:54:30	03:55:30	BL06005	O114

Table 3.3: A table representing the different types of anomalies found in the data set and removed during the phase 2 clean up process. Examples are extracted from 16 November 2015

The first anomaly in Table 3.3 is jobs that have duplicate **ORIG_LOC** and **DEST_LOC** locations. These anomalies show that a pallet is removed from a specific rack location and placed in the same location moments later. Table 3.12 shows jobs classified with this anomaly type make up 0.98% of the dataset. After consulting with PEP it is determined that these are caused by human error and can be excluded from the dataset.

Anomaly 2 are jobs with identical **PALLET_ID** entries but are stored from different **ORIG_LOC**. Here the same pallet is stored from different locations to the same storing destination. Referring to Table 3.3 a pallet with ID 3523574 is moved from three different origin location to the same storing location AD03020. After consultation it is determined that this anomaly is in fact the route of a pallet until it is stored in the storage rack system. Pallet 3523574 moved from the storage location MAR129 to the temporary floor storing location PSG002. From here the pallet is moved to the storage queue at the end of aisle AD from where a high lift stores it in AD03020. The movement of the pallet between location MAR129 and PSG002 does not happen within the storage rack area and thus falls outside the scope of this study and can be excluded from the

dataset. This type of anomaly comprises (246 811) 52.35% of the dataset as shown in Table 3.12.

The third type of anomaly, divided into anomaly 3a and 3b, make up (5 209) 1.10% of the data set as shown in Table 3.12. Anomaly 3a shows three different pallets being stored to the same storage location AY01013. The data set show that pallet 3519031 is stored and moments later a new pallet 3519032 is stored in the same location. Pallet 3519031 is not removed from the storage location before the next one is stored there. This means that according to the data set three different pallets are stored on top of each other in the same location. Anomaly 3b indicate the inverse where three different pallets are retrieved from the same rack location BG01028. The data set shows that after pallet 3318647 is removed pallet 3318644 is never stored in the location before it is removed. It is believed that these anomalies are caused by individual boxes being removed from or placed upon a pallet and the format the WMS is logging it, makes it seem as if an entire pallet is being removed or stored.

It is only possible to identify these anomalies when there is more than one job logged, making it impossible to be sure if any of the other jobs may fit the same criteria. Three possible methods are investigated in an attempt to manage this type of anomaly. The first method is to ignore all jobs prior to the last recorded job when grouping the anomalies. This method assumes then that the last entry is a storage or retrieval job where the pallet is moved. This will cause a rack location to become available or occupied depending on the movement type performed. The second method is to completely ignore the jobs that falls within this anomaly type. The third method is a hybrid of the first two. This methods takes the logged time for a job to be completed into account when determining which jobs should be ignored. Using this method the completion times for each job is determined. If the time is unrealistic the job is ignored. In the case that a job has an realistic time it is handled as a normal job and remains in the data set. The third method is used since it decreases the accuracy of the data the least, while still following the logic of the high lifts.

Anomaly 4 represents the cases where a job has exact duplicates in the data set. Table 3.3 shows three jobs with identical entries in all fields. The fact that these jobs are exact duplicates means that the WMS logged the same job multiple times. The reason why this happens could not be determined through consultation with PEP management, but it was decided to be an ignorable event. When this anomaly is encountered in the data set all but one of the entities are deleted. Table 3.12 shows (2 931) 0.62% of the data points are of this type.

After all the anomalies have been removed, the cleaned data set is used to construct a job list that is used in the simulation. A total of 211 910 data entries remain after removing the anomalies. The data does not give the high lift travel speeds or information regarding which high lift is used for each job. Using the data set the high lift movement speeds are determined using a job's starting and ending time.

3.4 High lift movement time data

For a simulation to have meaningful and credible results it is crucial that the simulated movement speeds and times accurately represent that of the physically observed times. An analysis of the start and end times of each job is performed to ensure that each simulated high lift move represents real life at an accurate velocity.

3.4.1 Storing and retrieving times

When calculating the movement speed of a high lift the travel times are needed. The travel time consists of the time a high lift needs to move between a queue location and rack location. The data points in the data set, received from PEP, contain time stamps that included the time at which the job is scanned to start and the time at which the job was completed. Using these time stamps it is possible to calculate the time it took to complete a job. An analysis of the time stamps exposes more anomalies present in the data set. It is unclear if these anomalies are caused by logging errors or if it is due to a common accruing event by just investigating the data set. Table 3.4 shows three anomalies encountered in the data set when the start and end times are used to calculate the completion time for a job. Storage and retrieval jobs are considered separately due to the possibility that different direction of movement, in or out of the storage rack, effect the travel times differently.

Case	MOVE_TYPE	START_TIME	END_TIME	TIME_DIFF	ORIG_LOC	DEST_LOC
1	LINEBUILD	01:25:43	01:25:45	00:00:02	AF01001	P124
	STORAGE	08:55:02	08:55:05	00:00:03	PSG013	AY04031
	MOVEMENT	02:06:29	02:06:32	00:00:03	AC05010	AD01032
2	LINEBUILD	08:57:54	08:58:54	00:00:47	MAR123	BE05029
	STORAGE	13:43:08	13:43:47	00:00:39	AG05012	G122
3	LINEBUILD	15:44:48	15:58:40	00:13:52	PSG013	AY04002
	STORAGE	02:12:59	02:39:39	00:26:40	BI04021	L132

Table 3.4: A table displaying examples of time differences found in the data categorised according to three cases. Case 1 and 2 illustrate the data points removed in the phase 3 clean up process.

It is necessary to understand the scanning process before going into detail regarding the time stamps in Table 3.4. In the event of a pallet being picked up from a queue the high lift operator drives past the pallets before picking it up. This is done so that the high lift operator is able to reach and scan the pallet's bar code on the side of the pallet. After the pallet is scanned the high lift operator moves backwards until the high lift's fork is properly aligned with the pallet. The fork extends perpendicular to the facing direction of the high lift, moving underneath the pallet. The pallet is picked up and the operator moves it to the assigned location in the storage rack. When the high lift reaches its destination the operator scans a bar code on the side of the storage rack. The pallet is then stored in the allocated storage location. During a retrieval movement the high lift moves to the pallet's location and the operator scans the bar code on the pallet and rack location before retrieving the pallet. After the pallet is retrieved it is moved from the rack location to the retrieval queue. After the pallet is placed in the queue and the fork has been removed from under the pallet the high lift operator drives past the pallet to scan the bar code on it.

Comparing the operation logic with the time stamps recorded shows that the WMS does not record the entire movement. The time between when a high lift arrives at a rack location, after the location is scanned, and the pallet is stored is excluded from the time stamps. Table 3.5 summarises the time, manually recorded at PEP's DC, it takes a high lift from when it arrives at the queue or storage location until the high lift unloaded or picked up a pallet. Using Table 3.5 it is possible to determine the least amount of time needed to complete a movement type. When a high lift does a storage job the action of picking a pallet up from the queue takes at least 11.70 seconds. This means that even if the high lift is able to instantly move to its destination the job

would take a minimum of 11.70 seconds to complete. Further more, since the storage job time stamps do not include the action of inserting the pallet into the rack location a minimum of 10.55 seconds must be added to all storage jobs. The time stamps of retrieval jobs include both the time for removing a pallet from a rack location, minimum of 9.52 seconds, as well as the time to place a pallet in a queue, minimum of 10.99 seconds. Storage and retrieval times stamps also include the time between the queue and storage location. This means that when looking at the time stamps a storage job completed under 11.70 seconds and a retrieval job completed under 20.51 seconds is impossible. All time stamps where the travel time is less than these times are excluded when analysing the high lift movement speeds. Time stamps meeting this criteria form part of case 1 in Table 3.4 and make up (66 383) 14.08% of the phase 1 data set and (22 377) 10.56% of the phase 2 data set.

	Queue(in seconds)		Storage location(in seconds)	
	Remove	Place	Remove	Place
Min	11.70	10.99	9.52	10.55
Max	39.59	30.87	23.18	35.00
Average	23.16	17.89	13.41	18.94

Table 3.5: Table representing the time (in seconds) it takes a high lift to pick up and put down pallets.

Case 2 in Table 3.4 exhibits time stamps that are regarded as accurate/usable travelling times for the different high lift actions. Data points that fit in this category are used to construct the phase 3 data set.

Case 3 in Table 3.4 refers to time stamps that are flagged as outliers by the box plot method [33]. Using the data set after phase 1 and phase 2 clean up is applied Case 3 is satisfied for (29 372) 6.23% and (13 992) 6.57% of the data points respectively. Each storage location is inspected separately to determine its outliers concerning job completion times. All outlier time stamps are ignored when analysing the high lift movement speed due to it not being realistic when compared to observed time stamps.

3.4.2 Analysis of the high lift travel times

Statistical techniques are applied to the historical data sets to achieve a better understanding of the high lift movements. These techniques include calculating the expected travel times of a high lift to determine the historical data's usability. Further, the effect that cleaning the data had on the expected travel times to each rack location is investigated. Finally, an analysis is performed on the data in an attempt to find a method that is able to accurately generate simulated travel times to and from each rack location.

The high lifts used in PEP's Durban DC closely resembles the EKX 410 Electric order picker/tri-lateral stacker [13]. Using the specifications of the EKX 410 it is possible to calculate the expected travel times between queues and rack locations. The EKX 410 manual states that this high lift is capable of moving horizontally at a velocity of 2.5 m.s^{-1} and a vertical velocity of 0.4 m.s^{-1} [13]. The general dimensions of a pallet are used, $1200 \times 1200 \times 150 \text{ mm}$. Assuming that a high lift is only capable of either moving horizontally or vertically equation (3.1) is formulated to calculate the expected travel time (T).

$$T = (P_W \times C)/V_h + (P_H \times L)/V_v \quad (3.1)$$

The variables used in equation (3.1) are defined as the pallet width in millimetres (P_W), the number of rack locations moved horizontally (C), horizontal velocity (V_h), pallet height in millimetres (P_H), the number of rack locations moved vertically (L) and the vertical velocity (V_v). Because the lifting and lower speeds (0.4 m.s^{-1}) are the same the travel times are the same irrespective of the job type being performed. Using equation (3.1) the expected storage and retrieval travel times are calculated and are plotted in Figure B.1. Before the expected travel times can be used to validate the usability of the historical data the effect of cleaning the data needs further analysis.

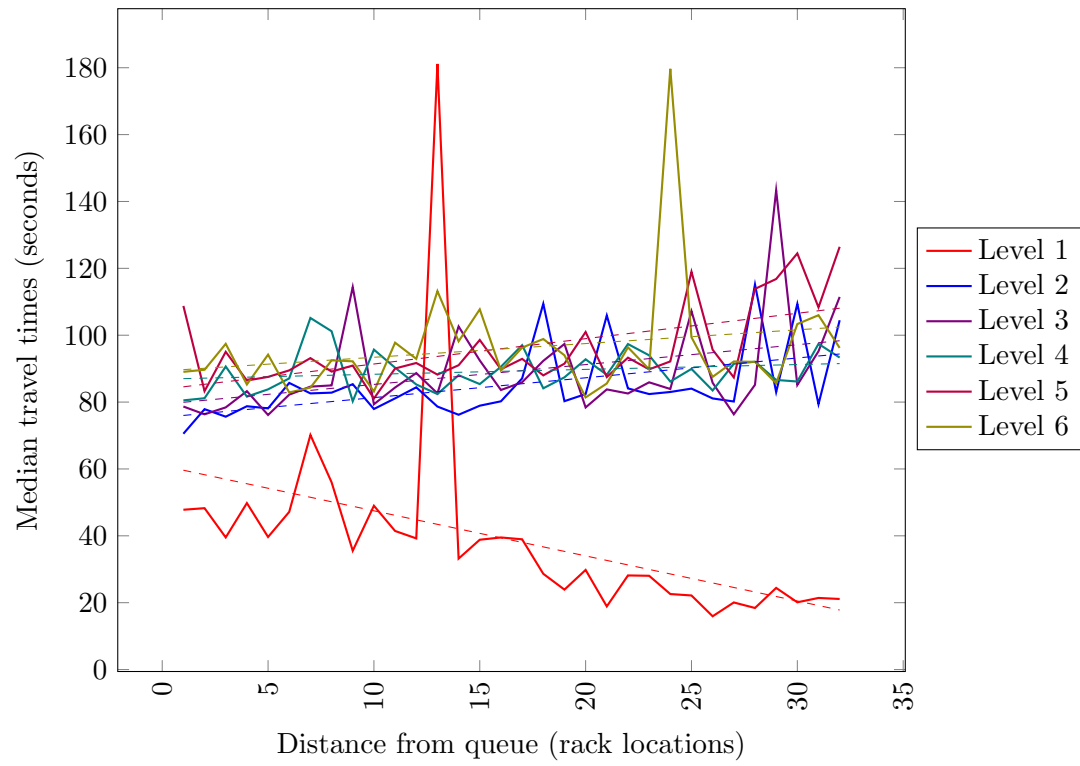
Before the historical data can be used it must first be shown that the historical data, in its current state, accurately portrays the expected movement and travel times of the high lifts. The historical data is investigated by applying the phase 1 clean up. In an attempt to better understand the difference between storage and retrieval travel times they are handled separately.

Figure 3.1 illustrates the correlation between the median travel times, in seconds, and the travel distance, in rack locations, for each rack level. The median travel times are calculated for the storage and retrieval movement types after the phase 1 clean up process is applied to the data set.

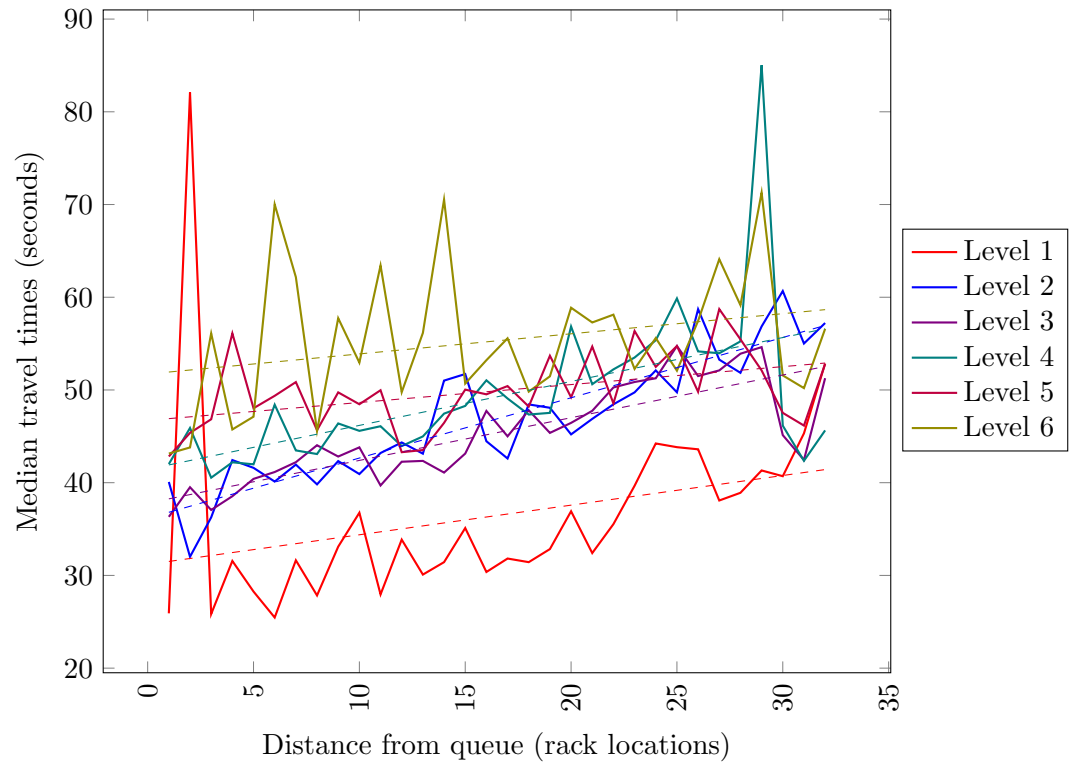
Regression lines are added to each storage rack level, illustrated with dashed lines, to better understand them. Linear regression lines are used as it fits the distribution of data the closest. Linear regression lines are useful when summarising the dependence of one variable on another [8]. The coefficient of determination (R^2) value indicates the variation, in data, between the two variables that have been plotted. A high R^2 value means the data is spread tightly around the regression line indicating more variance is accounted for by the regression model between the variables. A low R^2 value means the data is spread loosely around the regression line and indicating little of variance is accounted for by the regression model between the variables. The closer the data point are to the regression line (high R^2 value) the more accurate the predictions are when generating travel times. According to Chin [7] data with a R^2 value higher than 0.67 are classified as having a substantial usefulness while a R^2 value of less than 0.19 indicates a low level of usefulness.

It is expected that for storage jobs the travel time increases as the distance from the queue to the storage rack location increases. The regression lines in Figure 3.1(a) shows that storage jobs performed on Level 1 do not follow the same pattern as the other five levels. The travel time decreases as the distance increases. It is unclear why the travel times for this level does not follow the same trend as the rest. The most likely explanation is that Level 1 is mainly used for new stores and single carton storage, the same as with level 6, but this is not confirmed. Another explanation for the unexpected distribution is that Level 1 does not have sufficient data points per rack location to give an accurate line fit, resulting in less accurate median travel times being calculated, shown in Figure 3.2. It is also shown that the rack locations closest to the queue have far less data points compared to the cell locations furthest away from the queue. This also contributes to medians being inaccurately calculated making the travel times for Level 1 to appear irregular. Because it cannot be confirmed why the data of Level 1 is irregular these data points cannot be ignored and is handled as is.

Visually comparing the regression lines with the expected distribution (Figure B.1) it is clear that they do not resemble each other. The lower level regression lines tend to cross the higher level regression lines, indicating that the vertical distance affects the travel times less as the vertical distance increases. The regression line for Level 4 shows the travel time as being uniform over the storage rack. This means that for Level 4 the travel time is independent of the horizontal distance. Comparing the regression line of Level 4 with the other regression lines shows this is the only regression line with this characteristic making it unlikely to be accurate.



(a) Storage times.



(b) Retrieval times.

Figure 3.1: Historical data set's median travel times, in seconds, to each storage location for the different levels using after all phase 1 clean ups are applied.

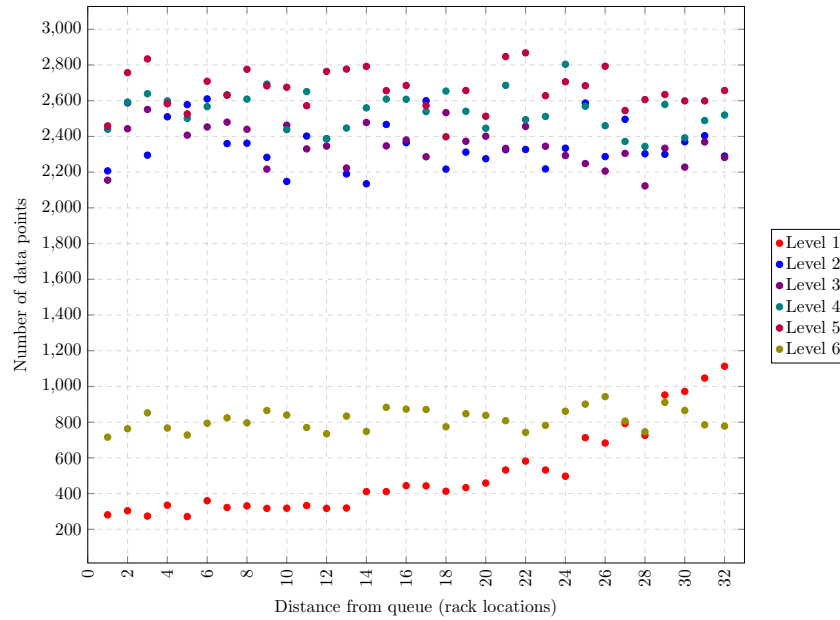


Figure 3.2: A visual representation of the total number of storage times stamps available at each distance from the queue, in rack location, per level.

Inspecting the regression line equation for each level, in Table B.1, indicates that the travel times are less dependant on the horizontal distance than expected. Further more the effect of the horizontal distance fluctuates depending on the level. Calculating the R^2 values, shown in Table 3.6, for Figure 3.1(a) indicates that the data points are very loosely scattered around the regression lines and almost no correlation between travel time and movement distance exists for any level. Only Level 2 (0.257) and Level 5 (0.347) have R^2 values above 0.19 meaning the storage data set, in its current state, does not accurately portray the expected storage movements.

	Phase 1 storage data	Phase 1 retrieval data
Level 1	0.190	0.080
Level 2	0.257	0.810
Level 3	0.159	0.681
Level 4	0.048	0.295
Level 5	0.347	0.205
Level 6	0.051	0.078

Table 3.6: Table summarising the R^2 values for the regression lines of the median travel times to each storage location using the historical data after the phase 1 clean up.

Figure 3.1(b) illustrates the median travel times, in seconds, between each rack location and the retrieval queue for the different levels. It is expected that as the high lift retrieves pallets further away from the retrieval queue the travel times increase. Adding regression lines shows that the data follows the expected pattern more closely than the storage data. Inspecting the regression line equations (Table B.1) it is noted that the horizontal distance has little effect on the travel times. The effect the horizontal distance has on travel time fluctuates depending on the level. Regression lines also cross each other which are not supposed to be the case. The coefficient of determination in Table 3.6 indicate strong correlation between the travel time and horizontal distance for the retrieval jobs performed on Level 2 (0.810) and Level 3 (0.681). The R^2 values for the remaining levels, however, indicate there is little correlation between travel time and

horizontal distance with data points scattered loosely around the regression lines. This means that using the retrieval travel time data, in its current state, is unable to accurately portray the expected retrieval movements. Further clean up of the data is required. Analysis is thus performed on the data set after it has been cleaned, all anomalies are removed, to determine if it can be used to generate accurate travel times.

Figure 3.3 depicts the relationship between travel distance, measured in rack locations, and median travel times, in seconds, for storage and retrieval jobs. The median travel times are calculated using the data set after the phase 1, 2 and 3 clean ups are performed.

Inspecting the median storage travel times, in seconds, in Figure 3.3(a) the regression lines for Level 2–6 closely resemble the distributions of the expected travel time (Figure B.1). The clean up process merely reduced the number of data points, per rack location, for all levels. It is thus expected that the regression line of Level 1 still does not follow the same pattern as the other levels. Unlike with Figure 3.1(a) the regression line equations (Table B.1) for Figure 3.3(a) show that the travel times are equally affected by the horizontal distance irrespective of the level. The regression lines show a distinct increase in travel time as the level increases indicating that the travel times are dependant on the vertical distance. According to Table 3.7 all the R^2 values are calculated to be above 0.67. This means that the data has a low variability around the regression lines, indicating the cleaned storage data is a good fit.

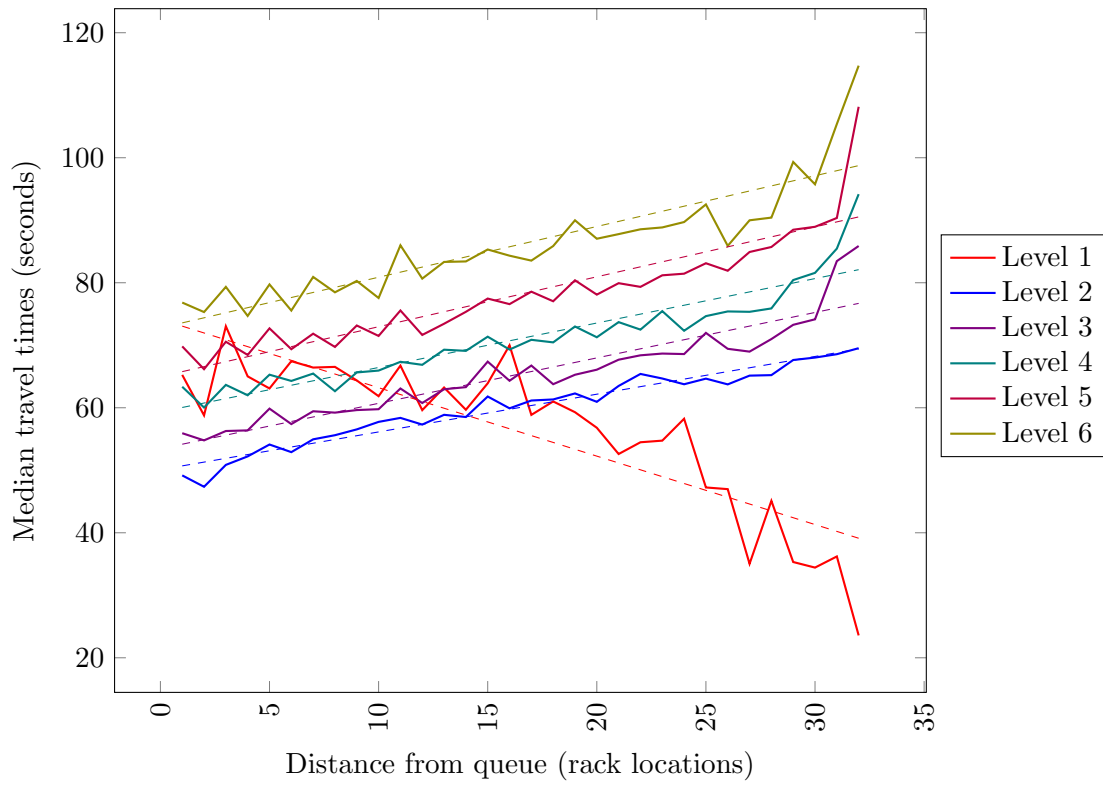
Figure 3.3(b) illustrates the median travel times, in seconds, for retrieval jobs after the data set is cleaned. The cleaned data set improves the shape of the regression lines to better align with the expected distribution shown in Figure B.1. Table 3.7 shows that the R^2 values for all the levels are above 0.67 indicating a low variability. This means that the median travel times for retrieval jobs are a good fit.

	Phase 3 storage data (R^2)	Phase 3 retrieval data (R^2)
Level 1	0.720	0.770
Level 2	0.948	0.933
Level 3	0.801	0.953
Level 4	0.839	0.903
Level 5	0.798	0.797
Level 6	0.762	0.819

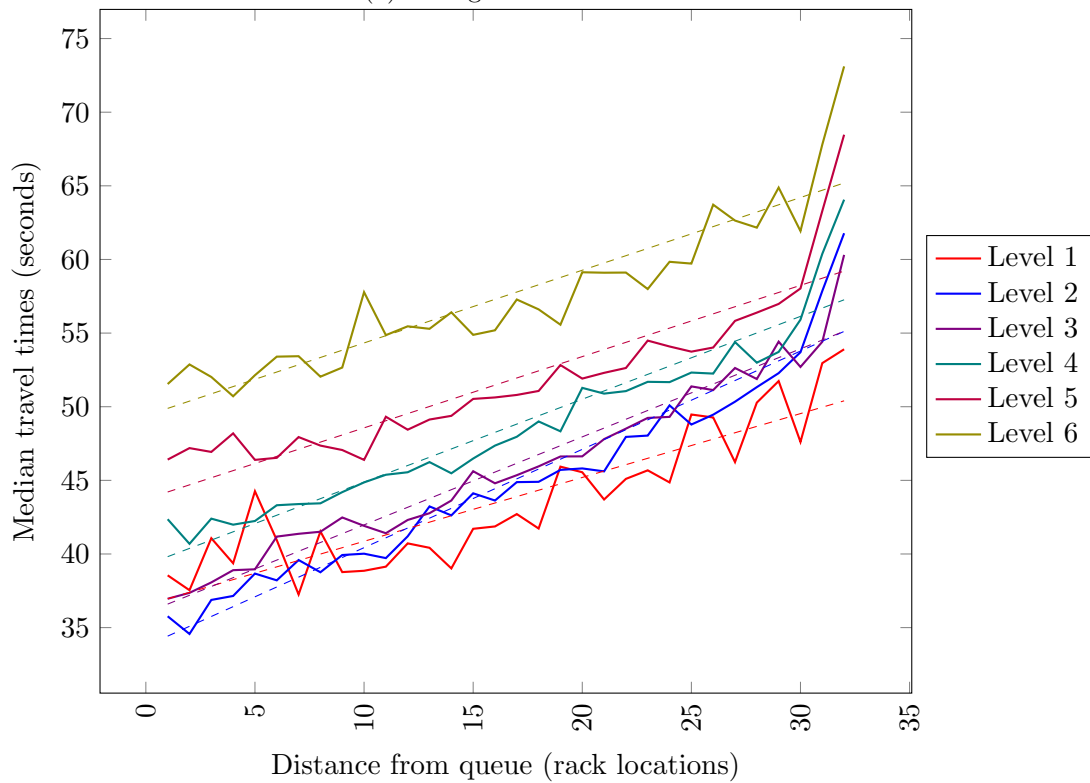
Table 3.7: Table summarising the R^2 values for the regression lines of the median travel times to each storage location using the historical data after the phase 1, 2 and 3 clean ups.

Analysis of the data set reveals that the data set in its original state is not sufficient to accurately generate travel times. Using the data set in its original form produces travel times with too high variability making them unreliable. Removing the anomalies and unrealistic time stamps from the data set, however, produces travel times that have a low variability. The cleaned data set further closely resembles the expected travel times (Figure B.1). However, a good fit does not ensure that accurate travel times can be generated using the cleaned historical data set. To prove that the historical data represents the high lift movement travel times accurately and can be used to estimate travel times the F-test ($\alpha = 0.01$) is used. The null hypothesis is that the historical data has the same distribution as the expected travel times. The alternative hypothesis is that the historical data does not have the same distribution as the expected travel times. The Hypothesis is tested for the storage and retrieval data sets for each level. The calculated p-values are summarised in Table 3.8.

Table 3.8 shows that the null hypothesis cannot be rejected for Level 2 – 6 but is rejected for Level 1, as is expected. Since the null hypothesis cannot be rejected for Level 2 – 6 the cleaned data sets can be used to generate travel times, in seconds, for these levels.



(a) Storage times.



(b) Retrieval times.

Figure 3.3: Historical data set's median travel times to each storage location for the different levels using after all phase 2 clean ups are applied.

	Phase 3 storage data (p-value)	Phase 3 retrieval data (p-value)
Level 1	0	0
Level 2	0.440	0.422
Level 3	0.339	0.445
Level 4	0.314	0.254
Level 5	0.266	0.262
Level 6	0.220	0.288

Table 3.8: Table summarising the p-values calculated using the F-test between the cleaned historical data and expected travel times.

If the regression line equations from the graphs are used constant travel times to each rack location is estimated. The travel times, however, are used in a simulation and using constant travel times as input would not give the desired results. The travel time distribution to each rack locations is investigated in an attempt to find a way to generate travel times with more variance.

3.5 Finding travel times

Methods for calculating high lift travel times, for the different movement types, are required. Due to the similarities between storage and retrieval movement types a single method is inspected. This method must be capable of simulating accurate travel times for both movement types between rack locations and queues. A method is also required that is capable of calculating the travel times relocation movements.

3.5.1 Travel times between rack locations and queues

Using the cleaned data set the distributions of seven cell locations are selected. The travel time distribution of each individual cell location is tested to determine if it fits a known statistical distribution. Cell locations are selected at random uniformly over the entire storage rack. Rack locations 2018, 2025, 3012, 3020, 4028, 5005 and 6016 are selected as the sample data sets. Figure 3.4 illustrates the distribution of rack location 4028 with bin sizes being determined using the Freedman–Diaconis rule [37]. Each rack location's travel times are imported into Minitab 17 statistical software [22] and its distribution is tested. The distributions tested are the Normal, Log-normal, Exponential, Weibull, Gamma and Logistic distribution. The process is repeated for both the storage and retrieval travel times. The null hypothesis is that the data fits any of the known distributions for each rack location. The alternative hypothesis is that the data does not fit to the known distributions for each cell location.

	2018	2025	3012	3020	4028	5005	6016
Normal	<0.005	<0.005	<0.005	<0.005	<0.005	<0.005	<0.005
Lognormal	0.03	0.025	<0.005	<0.005	<0.005	<0.005	0.073
Exponential	<0.003	<0.003	<0.003	<0.003	<0.003	<0.003	<0.003
Weibull	<0.010	<0.010	<0.010	<0.010	<0.010	<0.010	<0.010
Gamma	<0.005	0.044	<0.005	<0.005	0.023	<0.005	<0.005
Logistic	<0.005	<0.005	<0.005	<0.005	<0.005	<0.005	<0.005

Table 3.9: A table summarising the p-values calculated for each cell location for the known distribution using Minitab 17. The column headings represent the cell locations.

The null hypothesis is only rejected if the calculated p-value is smaller than the statistical significance level. Further more the smaller the p-value for the less likely the data is to follow the tested distribution. Testing the null hypothesis with a statistical significance level $\alpha = 0.01$ Table 3.9 shows that only three (37.5%) of the data sets do not reject the null hypothesis. It is assumed that if there is no distribution able to fit all eight data sets none of these distributions will be able to fit the data sets of the remaining rack locations.

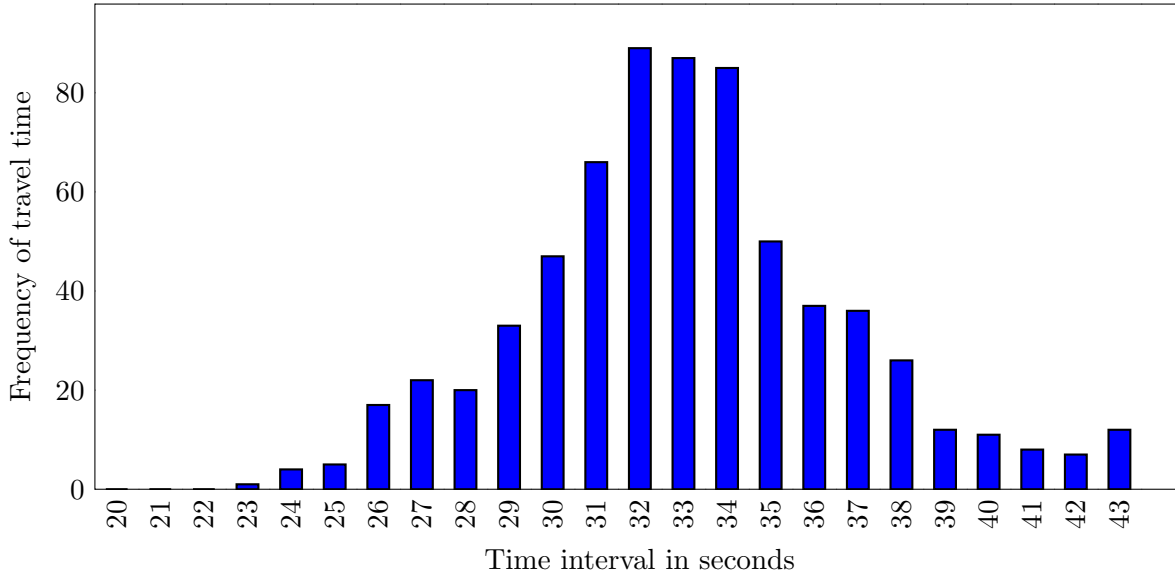


Figure 3.4: Graphical representation of a frequency graph for the different travel times for rack location 4028.

The fact that none of these distributions fits all of the rack location data sets an alternative method is needed to generate times stamps. Having 192 different distributions makes it impossible to find a single equation able to generate travel times. Instead a number generator is constructed that aims to replicate the historical data's distribution. Figure 3.5 shows the logic flow of how the number generator estimates a travel time from a rack locations historical data. The historical data is converted into a histogram with bin sizes of 1 second, to preserve accuracy. Each bin is then assigned a cumulative percentage value based on the number of data points within a bin out of the total number of data points. A random number is generated uniformly over the interval $[0,1]$ and is used to select a bin from the histogram. The bin value is returned as the generated travel time for the specific rack location.

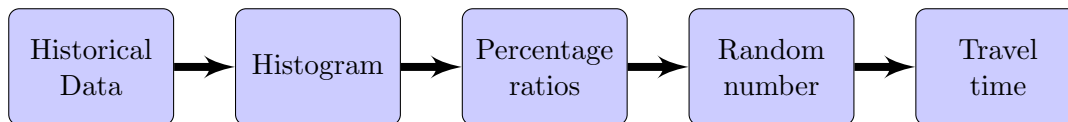


Figure 3.5: Illustration of how a storage rack location's distribution is used to generate a travel time.

Using this method it is possible to generate travel times, in seconds, between queues and any rack location for both storage and retrieval jobs. The more travel times are generated using the method the closer the generated times resembles the historical data, this can be seen in § B.1.2. Due to the lack of data points recorded for relocation movement types this method cannot accurately generate travel times for relocation jobs. An alternative method of calculating the travel times between rack locations is needed.

3.5.2 Travel times between two rack locations

The triangle distribution is a well known substitute in simulations when an input distribution is unknown [18]. The three parameters required to fit a triangle distribution are the minimum value, the maximum value and the value that resembles the expected value. It is not possible to determine these parameters directly from the data and needs to be estimated.

The first method discussed is to find a method capable of estimating the value that resembles the expected data value. For this method both the storage and retrieval movement types are used in unison to estimation the relocation travel times. The time stamps are categorised according to the distance between queues and rack locations. This means that the time stamps of storage jobs being stored in rack location 1032 and the time stamps of pallets retrieved from rack location 1001 are grouped together. Both the jobs performed on these rack locations have a distance of one rack location between the queue and the rack location. All outliers are removed from each rack location grouping's time stamps. The outliers are removed using the box plot method with a total of 1004 (0.6%) time stamps being classified as outliers over all rack locations. Using the newly formed data set the median travel times to each cell is plotted (Figure 3.6) for each level.

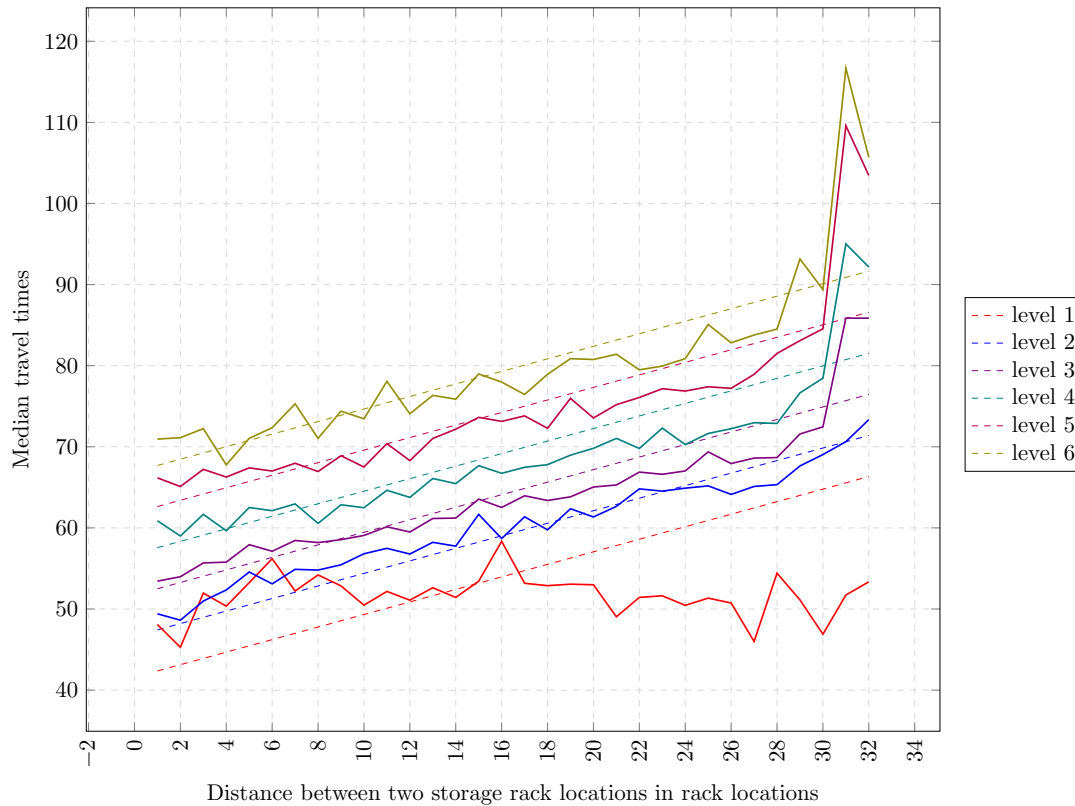


Figure 3.6: Estimated average travel times between rack locations using the Equation 3.2. The solid lines represent the median travel times of the combined data sets and the dashed lines represent the estimated data using Equation 3.2

It is expected that the further two rack locations are apart the longer the travel times become. Figure 3.6 shows that the median travel times of Level 1 does not follow the expected distribution. This renders the data from Level 1 unusable meaning Level 2 – 6 must be used to formulate an equation capable of estimating the expected data value for the triangle distribution. Although it is possible to estimate the parameter value using the regression line equations there would be six equations, Table B.3. The linear regression lines also will not fit the historical data very

well when looking at the R^2 values (Table B.3). Furthermore, the estimates generated for Level 1 would not fit the expected pattern. A single equation should thus be formulated that fits the historical data at least as good as the regression lines of Level 2 – 6. This equation should also be capable of accurately estimating travel times for Level 1.

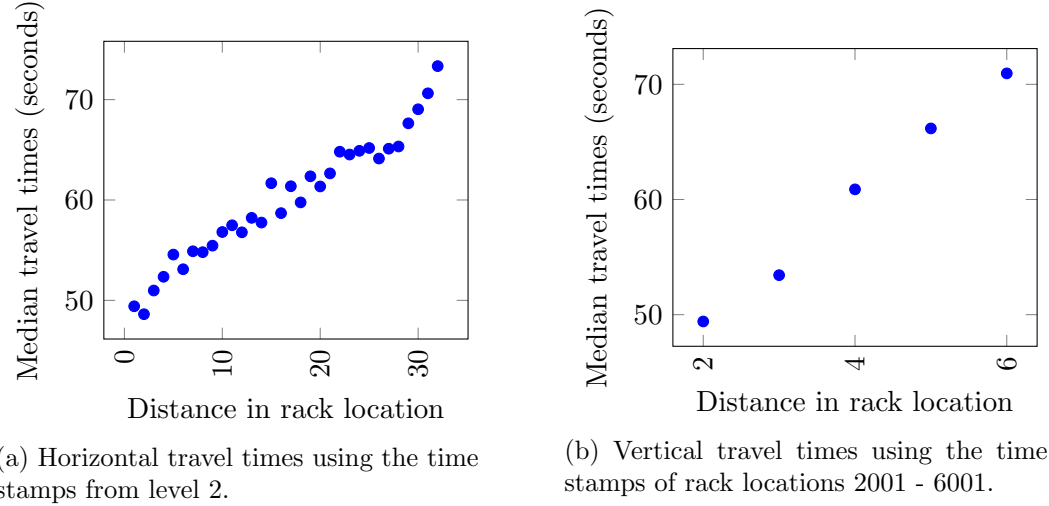


Figure 3.7: Illustrations of the travel time distribution focussing independently on the horizontal and vertical travel times receptively.

In Figure 3.7 the median travel times, in seconds, are plotted separately against the horizontal and vertical distances, in rack locations. Using a multivariate regression model a single equation is formulated. Multivariate regression is a method that estimates an outcome by investigating the relationship between multiple variables. The multivariate regression model is used to investigate the effect horizontal and vertical distances have towards the travel times. The multivariate regression model is performed using the median travel times to all rack locations (160) excluding Level 1. This equation resembles the historical travel times closely with a R^2 value of 0.84, Table B.3, indicating low variability. The expected travel times are estimated using the equation

$$T = 0.77x_h + 5.07x_v + 41.59, \quad (3.2)$$

where T represents the estimated travel time, in seconds. The variable x_h denotes the number of rack locations travelled horizontally between the two rack locations. The x_v variable denotes the number of vertical rack locations travelled between two rack locations. The coefficients in equation (3.2) show that the estimated travel times are relatively independent of the horizontal travel distance. The vertical distance, however, effects the travel times 5.58 times more than the horizontal distance for each equation.

Using equation (3.2) the median travel time to move any horizontal and vertical distance is estimated and plotted in Figure 3.6, as the dashed lines. The homogeneity of variance is tested between the estimated travel times, from the equation, and that of the historical median travel times, depicted in Figure 3.6. Using Levene's tests the null hypothesis that the variance between the estimated travels times and historical travel times are equal is assessed. Testing the null hypothesis with a statistical significance level $\alpha = 0.01$ it is shown in Table 3.10 that the null hypothesis can not be rejected for any of the tested levels.

From the results in Table 3.10 it is assumed that using equation (3.2) is capable of estimating realistic travel times for relocations performed on Level 1. Further the coefficient of determination between the estimated median travel times and median of the historical travel times are

		Level 2	Level 3	Level 4	Level 5	Level 6
Eq. (3.2)	Levene	0.22	0.76	0.89	0.49	0.42

Table 3.10: Table displaying the p-values obtained when testing the null hypothesis between the estimated travel times, in seconds, and the historical travel times, in seconds.

calculated in Table 3.11. The R^2 value for Level 1 shows none of the variation between the estimated and historical median travel times can be explained, as expected. The value of R^2 of the other levels, however, further verifies that equation (3.2) fits the historical median travel times well. It is thus possible to say that equation (3.2) can be used to determine the value that resembles the expected value.

	R^2
level 1	0.048
level 2	0.978
level 3	0.907
level 4	0.865
level 5	0.817
level 6	0.814

Table 3.11: Coefficient of determination, R^2 , between estimated travel times using Equation (3.2) and the historical travel times.

It now remains to find a method capable of estimating the minimum and maximum travel times for the triangle distribution. By repeating the steps followed to determine the parameter for expected travel time it is attempted to find an estimation for the minimum value. Formulating equations using linear regression lines, Table B.3, of the time stamps for the minimum travel times on Level 2 the R^2 values are calculated. It is shown by the R^2 that the estimated travel times generated using these equations will fit the historical data loosely. A multivariate regression model is thus imposed but this too will not be capable of accurately estimating minimum travel times. Table B.3 shows that none of the variation can be explained between the travel times and horizontal distance for minimum travel times. When selecting the parameter representing the minimum value it is thus chosen as the average of all the minimums from Table B.4 equating 21 seconds.

Finally, the parameter for the maximum travel time is determined by following the same steps. A multivariate linear regression model is used in accordance with the historical data and the travel times estimation equation is formulated as

$$T = 1.57x_h + 7.03x_v + 94.91, \quad (3.3)$$

where T represents the estimated travel time, in seconds. The variable x_h denotes the number of rack locations travelled horizontally between the two rack locations. The x_v variable denotes the number of vertical rack locations travelled between two rack locations. R^2 is calculated as 0.627 indicating a weak correlation between the travel time and movement distance. The weak correlation of Equation (3.3) means that travel times generated by it does not fit the historical maximum travel times well. The horizontal and vertical coefficient in equations (3.3) indicate that the maximum travel time are 3.05 times and 3.48 times more dependant on the vertical movement distance than the horizontal distance. Since the maximum travel times are relatively independent of travel distance, the average value of 140.35 seconds is calculated from the maximum times in Table B.4 and is used as the parameter of the triangle distribution.

3.6 Job list generation

When simulating the movement of pallets by means of high lifts in the storage racks it is important to have enough data points to sufficiently emulate the DC [36].

The job list is constructed using data points spanning over a five day period. It is required that the selected days are five consecutive work days, to allow a proper flow of pallets. If days are not selected consecutive it is possible for jobs to try placing pallets in occupied storage locations that were not made available due to the day not being included in the job list. The days are selected primarily by determining the five consecutive days with the most data points and need the least amount of clean up.

In order to select the best candidates for the days a month is selected first. It is important that the selected month has a sufficient number of jobs whilst the necessary clean-up is kept minimal. Using the information summarised in Table 3.12 the best candidates are selected with respect to the total remaining data points and clean up required (CU%). The months Mar-15 (22091), May-15 (19951) and Nov-15 (19037) are selected as possible candidates. The number of public holidays in March and May causes numerous days that have no jobs being done. In the case that there is an off day, public holidays and Sundays, the next day starts with the day shift. This means that a day after an off day consists of less jobs than a normal day. Thus a month with many public holidays have less consecutive days with evenly distributed job numbers. November had no public holidays, meaning more days have full shifts, and jobs are more evenly distributed between each day. Although Nov-15 does not have the most data points or the least amount of clean ups it has the most consistent job distribution and is thus selected.

Month	Total jobs	Storage	Retrieval	Movement	Anomaly 1	Anomaly 2	Anomaly 3	Anomaly 4	CU%	Remaining
Feb-15	10146	7788	2186	172	115	5306	107	61	55.09	4557
Mar-15	48448	36937	10759	752	452	25127	542	236	54.40	22091
Apr-15	38023	27687	8952	1384	443	18431	714	276	52.24	18159
May-15	40402	28654	8579	3169	755	18600	584	512	50.62	19951
Jun-15	31122	22775	7959	388	290	15252	443	244	52.15	14893
Jul-15	43458	34456	8545	457	350	23928	397	119	57.05	18664
Aug-15	37613	29457	7627	529	361	20880	417	199	58.11	15756
Sep-15	35848	27450	7671	727	324	19357	432	224	56.73	15511
Oct-15	43531	34082	8349	1100	446	23913	429	269	57.56	18474
Nov-15	42277	31960	9761	556	339	22325	295	281	54.97	19037
Dec-15	27836	20631	6900	305	183	13727	422	193	52.18	13311
Jan-16	41460	33292	7545	623	335	22964	246	121	57.08	17794
Feb-16	31303	24622	6287	394	213	17001	181	196	56.20	13712
Total	471467	359791	101120	10556	4606	246811	5209	2931		211910
%	—	76.31	21.45	2.24	0.98	52.35	1.10	0.62		44.95

Table 3.12: A table representing the number of jobs for each month. Included is a summary of the types of jobs and anomalies. The highlighted row represents the month's data set that is selected.

Using the data from Nov-15 six consecutive days are selected. The selection criteria for the days are the same as with the month selection. Consecutive days are defined as working days that follow each other. This means that Monday follows on Saturday since no jobs are performed on Sundays.

Using Table B.5 groups are created consisting of five consecutive days each as shown in Table 3.13. Sorting the information of Table 3.13 according to the number of jobs after the clean up is performed the top five groups are found to be 12(4653), 11(4534), 13(4497), 10(4400), 14(4221). Although groups like 1 need less clean-up compared the candidate groups, about 6.51% less, the number of jobs are too low to be considered. A group is selected to maximize the number of jobs to be simulated while minimizing the clean up needed. Group 12 is selected

as it has the most data entries with a relatively low number of clean ups needed.

Group	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Days	2-6	3-7	4-9	5-10	6-11	7-12	8-13	9-14	10-15	11-16	12-17	13-18	14-19	15-20	16-21	17-22	18-23	19-24	20-25	21-26	22-27
Jobs	6550	6618	7079	6946	7474	7856	8630	8720	9373	9912	10528	10670	10500	9722	9026	8792	8345	8417	8418	8078	8551
CU%	49.88%	54.11%	56.17%	57.13%	56.68%	56.35%	54.21%	53.59%	55.16%	55.61%	56.93%	56.39%	57.17%	56.58%	56.40%	54.15%	54.18%	52.68%	52.28%	52.33%	56.03%
Remaining	3283	3037	3103	2978	3238	3429	3952	4047	4203	4400	4534	4653	4497	4221	3935	4031	3824	3983	4017	3851	3760

Table 3.13: A table representing the number of jobs for each day including the number of anomalies.

Days 14-Nov-2015 to 19-Nov-2015 are selected to be used in the simulation as they require a relatively low amount of clean up while still containing a large number of data points. The selected data points can now be used to compile a job list. The data received from PEP does not indicate which high lift performed a job. As a final step it is required that it is determined which high lift did what job.

3.7 High lift assignment

During a shift it is not only the high lift operators who perform tasks in the storage racks. Workers storing and retrieving individual cartons are also logged by the WMS and included in the data set. This means that not all employee ID's logged in the data set refer to the high lift operators. A method of assigning high lifts to jobs are thus needed. In an attempt to assign high lifts to jobs the data is split into the day and night shifts according to the time a job is started (Table 3.14).

	Begin time	End time
Night shift	18:00:00	06:00:00
Day shift	07:00:00	17:00:00

Table 3.14: A table representing the time interval for each shift.

The number of jobs each employee completes in a shift is determined. The employees of each shift is sorted according to the number of jobs they completed from most jobs completed to the least. This is shown as an example in Table 3.15. The maximum number of high lifts present in the storage rack area is five. This means that if there are more than five unique employee ID's the same high lift must be assigned to multiple employees.

ID	Number of jobs	High lift number
1629	167	1
1934	151	2
1628	125	3
1623	109	4
1583	108	5
1657	43	5
1006	18	5
1952	3	5
1007	1	5

Table 3.15: Example of how the high lifts are assigned to employees for a day shift on 17-Nov-15.

Starting with the employee with the most completed jobs the first high lift is assigned, high lift 1. Moving down the list of employees the remaining high lifts are assigned. If high lift 5 is assigned but there are still employees remaining high lift 5 gets assigned to them as well, as

indicated in Table 3.15. This is done to ensure that jobs are more evenly spread between the high lifts. This process is repeated for all shift ranging from 14-Nov-2015 to 19-Nov-2015. Each job's assigned high lift is added to the job list completing the process of finding an usable job list to simulate.

3.8 RC Value assignment

For the placement algorithms to be executed properly all jobs need the have an RC value assigned to them. Table 3.16 shows an extract from the historical data where some jobs do not have a RC value assigned to them. The example show that the transition from a job with a RC value to a job without one looks the same as between two jobs with RC values. From the data there is thus no evidence that these jobs are handled differently from jobs that have RC values.

PALLET_ID	HIGH LIFT	START_TIME	END_TIME	START_LOC	END_LOC	SKU	RC
3521555	High lift 1	10:26	10:27	PND303	AC05016	116269	3
3521083	High lift 1	10:28	10:29	PND303	AC05004	281949	
3521084	High lift 1	10:30	10:31	PND303	AC05017	281947	
3521560	High lift 1	10:32	10:33	PND303	AC05021	116167	3
3521558	High lift 1	10:33	10:34	PND303	AC05020	116346	3

Table 3.16: Example extracted from 11-14-2015 showing storage jobs with undefined RC values in the historical job list.

A method is required to logically assign a RC value to these jobs. It is, however, unnecessary to determine RC values for retrieval jobs. Pallets of retrieval jobs are already in the storage racks and assigning RC values to them are redundant. Moving these pallets, too a different rack location, based on new RC values will increase completion time and distance - this contradicts the purpose of the algorithm. Because of this, RC values are only determined for storage jobs.

The historical data of November 2015 is used for this task as it is deemed the cleanest data (§ 3.6). Table 3.17 summarises the number of jobs, the historical data, that have RC values assigned to them. It also shows the number of jobs with no RC value assigned. It can be calculated that 51.36% of storage jobs do not have a RC value assigned to them - far to significant too be ignored.

RC value	1	2	3	4	Unassigned
Number of storage jobs	711	2069	1160	794	5031
Number of retrieval jobs	814	1891	1094	727	4746
Total Number of jobs	1525	3960	2254	1521	9777

Table 3.17: Summary of the number of storage jobs that each have a RC value assigned to them.

As a first method of assigning RC values, each job's historical data is considered. The historical data, however, contains no stock related information that can help identify RC values (type of stock, rate of sale). The only usable information is the date at which a pallet is stored or retrieved. The duration of time a pallet is stored, in the storage racks, is calculated by calculating the difference between the date at which it is stored and retrieved. A jobs's RC value is determined by the number of days the pallet is stored. A pallet stored for less than or equal to 7, 14, 21, 28 days is assigned a RC1, RC2, RC3, RC4 value respectively.

Considering only storage jobs with no RC values their time in storage is calculated and summarised in Table 3.18. RC values are assigned to these storage jobs based the number of days spent in storage. There are, however, still 3 614 jobs that do not have RC values after this method is applied. Because both the storage and retrieval date is required for this method it cannot be applied to pallets that are only stored or retrieved, in the November time frame. Without both these dates it is impossible to determine how long a pallet was stored.

Days stored	≤ 7	≤ 14	≤ 21	≤ 28
Number of jobs	1168	189	48	12
Percentage of jobs	82.43	13.34	3.39	0.85

Table 3.18: Summary of the number pallets that are stored in the storage racks according to the number of days a pallet was stored during November. Only pallets with no RC value assigned to them are considered.

The second method used to assign RC values is done by using the percentage in Table 3.18 as probabilities. Using the cumulative probability of each RC value occurring The remaining jobs, without RC values, are assigned a RC value at random based on the cumulative probability for each RC value.

After applying these two methods to all jobs the compilation of the job list's RC values changes. The new ratio for each RC value is summarised in Table 3.19. It can be calculated that 49.75% of the pallets that flow into the storage rack during the simulated time have RC values of one. Further 28.06% of the pallets that are stored are RC2 and RC3 comprises 13.62% of the pallets. The remaining 8.57% of the pallets have a RC value of four assigned to them.

Replenishment cycle	1	2	3	4	Unassigned
Number of jobs	4858	2740	1330	837	0

Table 3.19: Summary of the number of jobs that have each RC value assigned to them after the pallets with no RC value is altered.

CHAPTER 4

Simulation model

A simulation is the process of replicating a scenario based on a set of assumptions and mathematical models [44]. Simulations are used to investigate the effect changes would have on a system without altering the existing system [20]. When uncomplicated characteristics are used to compose a model it is often possible to obtain exact analytical results using mathematical methods. Most real-world problems are, however, so complex that an analytical approach cannot effectively evaluate a realistic model. These complex models are evaluated by means of simulations that numerically studies the model using computers and estimates an outcome solution for the model [19]. Simulating an accurately modelled scenario allows to isolate and identify the influence of parameters by changing their attributes [25].

4.1 Assumptions

Constructing simulation models that are able to exactly replicate real-world scenarios demand intensive computational power and mathematical models due to its complexity. It is thus necessary that assumptions are made, regarding the scenario, to reduce the complexity of the model while maintaining a high level of accuracy. The following assumptions are made regarding the simulation model for this study.

1. It is assumed that there are enough pallet jacks, due to their low labour, purchase, maintenance cost. Having the high lifts operate at maximum efficiency outweighs the cost of purchasing additional pallet jacks. Having enough pallet jacks means that the flow of pallets into and out of queues will always be sufficient, allowing high lifts to never have to wait. When the inflow of pallets are sufficient it means that while a high lift is performing storage jobs the required pallets are always available in the queue. Also if the outflow of pallets are sufficient it means that when a pallet is removed and needs to be stored in the queue there will always be an available storage location in the queue. This allows for any interaction between pallet jacks and high lifts to be ignored. A high number of pallet jacks would further keep the queues short keeping temporary storage areas from filling up and obstructing the movement of high lifts and pallet jacks.
2. During the course of a day no breakdowns occur — no data regarding break downs are available and can thus not be included in the model.
3. All pallets are handled equally by high lifts in terms of carry speeds, cautiousness and weight irrespective of stock characteristics. The received data has no information regarding

the type of stock stored on the pallets. It is thus impossible to find any correlation between the stock stored on a pallet and how the high lift operators handle the different types of pallets.

4. When moving from one aisle to another a high lift will never be obstructed or have to wait. High lifts are given the highest priority while moving in the storage rack area. This means that all other movement equipment will give way to high lifts.
5. High lift operators perform all assigned jobs precisely and efficiently. The correct rack locations are always selected without fault and no correction movements are ever necessary. Although it was observed that high lift operators sometimes do correctional movements the data does not distinguish between correctional movements and normal movements. No data can thus be extracted to accurately implement correctional movement into the model.
6. The flow of movement always stays constant. This means that irrespective of a pallet's characteristics it will always enter the storage racks from a storage queue and always be removed through the retrieval queues.
7. It occasionally happens that pallets are returned from the order picking area and stored back in the storage racks. The occurrence of this is so seldom that it is assumed not to happen.
8. Lastly all 192 rack locations under consideration are the same in terms of high lift interaction with them. All storage rack locations were constructed identically and for the same type of pallet. It is thus assumed that high lifts would interact with all in the same manner.

4.2 Simulation model

The programming language Python 3.4 is used to code a simulation model capable of solving the problem described in Chapter 2. Python is a programming language capable of coding an agent based simulation model according to the required specification without any boundaries imposed by custom simulation software. Using the built in libraries the model can further be expanded and refined as needed to replicate the real world system accurately. Finally, Python is completely open-source and free.

Figure 4.1 depicts the general process flow of the simulation model. The first process is importing external data used by the simulation. There are two main components when importing the data, namely the job lists and time distributions as discussed in Chapter 3. The imported time distributions can further be categorised into two smaller data sets namely the time distributions for removing a pallet from a rack location and the time distributions for storing a pallet in a rack location.

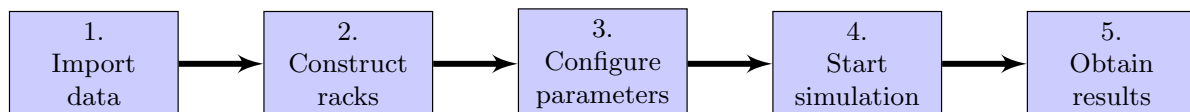


Figure 4.1: Flowchart depicting the process flow of the simulation model.

The second stage of the simulation model is to construct the storage rack system and place start up pallets. It is assumed that pallets that are removed from rack locations without being stored, according to the job list data, must have been stored prior to the simulated date. These pallets that are removed, but never stored, are thus stored in their respective rack locations before the

simulation starts as the start up pallets. The storage of these pallets do not contribute in any way to the output of the simulation. In the third stage the initial parameters are set for the the simulation model. This includes the selection of which movement logic and placement logic to use as well as whether or not the sequencing algorithm is used. The deadline parameter, in hours, is selected and is defined as how long after a job has been released should it be prioritised by the high lifts. The final parameters are the storage rack dimensions and simulation speed, where a speed of 1 means the simulation is run in real time. As the speed is increased the simulated time increase by multiples of the selected parameter. After all the initial preparations are completed the simulation may start.

The simulation is coded using multi-processing allowing sub processes to compile simultaneously. This allows for the simulation model to have a global process (parent process), this processes the information that is shared between high lifts like the configuration of the storage rack, movement times, simulation time and parameters. Each high lift is operated within a sub process (child process) that constantly returns information to the parent process regarding its actions. Having multiple processes allow for each process to function independently while still being aware of each other.

Figure 4.2 illustrates the process flow of a high lift sub process that forms the majority of the simulation program. The high lift sub process is executed and obtains all the initial information from the parent process.

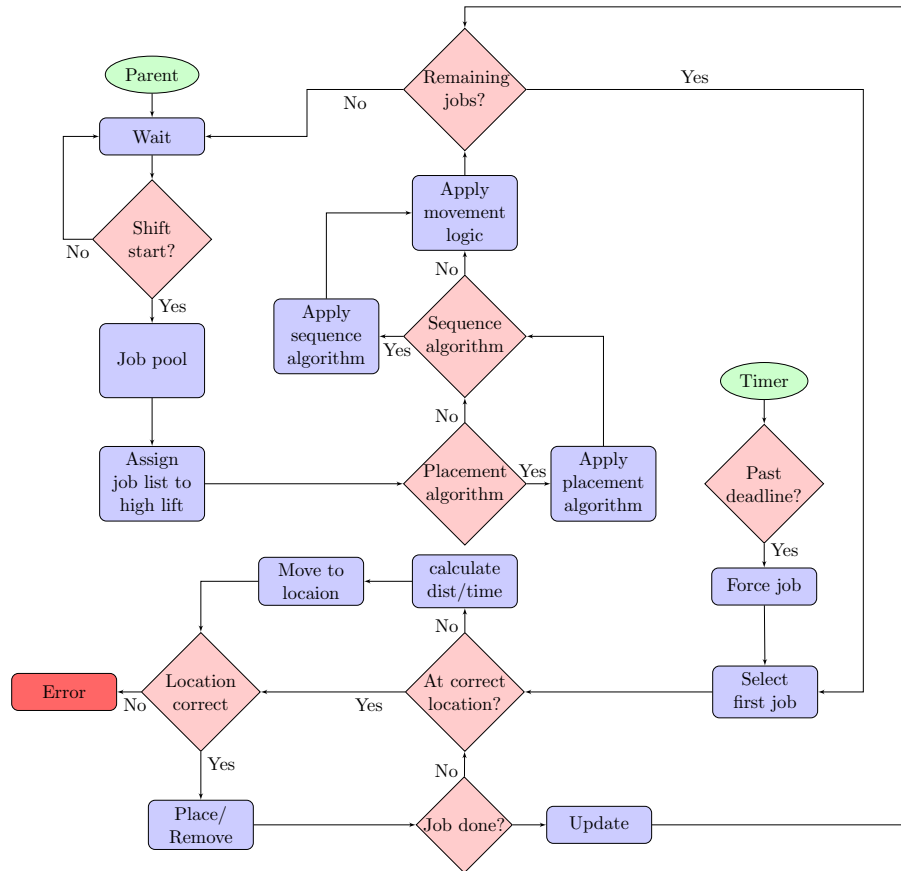


Figure 4.2: A flow chart illustrating the general flow and workings of the simulation model.

The process starts by waiting for the simulation time to reach or exceed the starting time of a shift. This is to ensure the simulation does not allow a high lift to operate while it did not in the real world scenario. The process compares the times after each simulated second until

the condition is satisfied. When a shift starts all jobs, for the shift, are released and stored in a job pool. Jobs remain in the pool until it is demanded by the assignment function. Jobs are released from the pool according to one of two methods. The first method is when simulating PEP's data exactly. With this method selected jobs are assigned to high lifts according to the historical data. Jobs are released when the simulated time reaches the starting time indicated in the historical data.

The next method starts when a proposed logic/algorithm is simulated. This method releases jobs according to PEP's storing logic, § [?], where high lifts start at the aisles on the edges moving to the centre. Jobs for a specific aisle will only be assigned to a high lift if the aisle and a high lift is available. In the case that an aisle is occupied the next aisle's jobs are assigned. If an aisle is skipped its jobs gets priority when jobs are assigned again. All jobs for a specific aisle are released at the same time. This is done to reduce the calculation times when applying the logics/algorithms to the job list. Instead of iterating through the entire job pool the simulation focusses only on the jobs applicable for the high lift's current aisle. After the jobs are assigned to a high lift their storage locations are determined (only if there are storage jobs).

Depending on the parameter the placement algorithm is applied to the job list. If one of the five placement algorithms (Rows, Columns, Steps, XY distance, Pythagoras) is selected the storage locations are changed for the storage jobs. After new placement locations are determined for the storage jobs or if no placement algorithm was selected the job list is transferred to the sequencing algorithm.

If it is selected to use the sequencing algorithm the Hungarian method is applied. The job list is altered a final time by changing the storage and retrieval queue locations depending on which of the five movement logics (SS, OS, SSP, OSP, CS) are selected.

In the case that there are no more available jobs a high lift will become idle and wait until new jobs are made available. High lifts are also seen as idle, in the simulation, when they are performing jobs in a different storage rack set. To prevent idle high lifts from occupying aisles they move back to the entrance of the storage rack system once they become idle.

If a high lift has jobs remaining in its current job list the first job is selected. After a job is selected it is determined if the high lift is at the pallet's final destination. If the high lift is not at the correct location the travel distance and times are calculated from its current location to the pallets final destination.

The distance is calculated by determining the shortest path between the high lift's current location and the destination. The simulation calculates the travel distance by either using Pythagoras or by summing the horizontal and vertical distances. Using Pythagoras the calculated distance resembles the real world scenario more accurately. However, when looking at the horizontal and vertical distances independently a better understanding of which part of a job contributed the most to dead heading is achieved.

When calculating the travel distance, in rack locations, only a high lift's current location and destination is required. The number of aisles that are travelled is calculated first. Each aisle consists of two storage racks e.g. AA and AB as they are on either side of an aisle. This means that a high lift only changes aisles if its current location and the destination racks are not part of the same pairing, shown in Table 4.1.

The number of aisle changes is multiplied by the width of an aisle resulting in the total distance needed for an aisle change. For movement within an aisle the distance is calculated by determining the number of horizontal and vertical rack locations between the high list's current location and the queue or rack location depending on the job type. This number of horizontal rack locations are multiplied by the width of a rack location to find the total horizontal distance.

Storage rack aisle pairings			
AA,AB	AC,AD	AE,AF	AG,AH
AI,AJ	AK,AL	AM,AN	AO,AP
AQ,AR	AS,AT	AU,AV	AW,AX
AY,AZ	BA,BB	BC,BD	BE,BF
BG,BH	BI,BJ	BK,BL	BM,BN
BO,BP	BQ,BR	BS,BT	

Table 4.1: A table consisting of the storage rack pairings per aisle.

The vertical distance is calculated by multiplying the number of vertical rack locations with the height of a single rack location. For travel distances to be validated the simulated distances and historical distances need to be exactly the same.

Travel times are determined using the equations and distribution methods described in Chapter 3. Depending on the job type the simulation uses one of three different methods to determine the travel time. If a storage job is performed the travel time distribution function is executed for the specified rack location. The function uses the distribution combined with a random number generator to estimate a travel time to the rack location as discussed in § 3.5.1. When calculating the travel times for retrieval the same function is used as with storage jobs except the retrieval distribution travel times are used. If the job requires the high lift to move a pallet from a rack location to another or an aisle change occurs the triangle distribution described in §3.5.2 is used.

No graphical user interface (GUI) is used when simulating the movement of all the high lifts. It is thus unnecessary to allocate process power to include the physical movement of a high lift. Instead the simulation is able to determine the simulated time of when a high lift enters or leaves an aisle as well as when a high lift picks up a pallet or stores a pallet. The movement of a high lift is split into the times at which it performs certain events, shown in Table 4.2. The sub process waits idle until the time of the next event is reached. At each event the high lift's contribution in distance, time and jobs completed are stored to the results.

Event	AA03015	Q-Retrieval	Q-Storage	AC02023	Wait
Time	09:33:17	09:33:54	09:34:38	09:35:07	09:35:50

Table 4.2: Example of how the movement of a high lift is split for the sub process and parent process to keep track of the high lift.

At each event the simulation confirms that the location is correctly configured. This means that when a pallet is stored the simulation verifies that the rack location is not occupied, disallowing multiple pallets to be stored in the same location. When a pallet is removed from a rack location the simulation verifies that a pallet was in fact stored in the rack location and that it is the specified pallet. The simulation determines if a stored pallet is the correct pallet by comparing the stored pallet's ID to the ID of the pallet in the job list. If there are any discrepancies between the rack location configurations and the job list, the simulation is terminated as it does not represent the real world scenario accurately any more. This conformation is essential to guarantee that the simulation model represents the real world scenario accurately. In case that no error is reported the pallet is stored/removed.

Depending on the high lift's current location multiple events might need to be reached before a job is completed. For storage jobs a job is completed after a pallet's storing event is completed. Retrieval jobs are only completed after the pallet is placed in a retrieval queue. After the job is completed the storage rack is updated to correctly represent the new rack configuration after pallets are stored and retrieved.

After each completed job the simulation determines if there are any remaining jobs in the high lift's current job list. If there are more jobs remaining they are completed otherwise the process waits until new jobs are released. The deadline timer sub process is an additional sub process that influences the high lift sub process. This process iterates through all the jobs every second comparing each job's start time with the simulated time. The deadline function is triggered when the simulated time exceeds a job's start time with the pre-described deadline parameter. A job that has triggered the deadline function is forced to the top of the high lift's job list that is closest to the job to ensure the job is completed as soon as possible.

The parent and sub processes run until all the jobs have been completed. The results of travel distance, travel times and jobs completed for the entire simulation and for each individual high lift is then saved to an Microsoft Excel file.

4.3 Verification and validation

Verification and validation is a fundamental component of simulation studies and without detailed verification and validation of a simulation model there are no justification to consider the results as credible [35]. The purpose of verification and validation is to confirm that the simulation model meets the predetermined specification, fulfils the intended purpose accurately and is credible and viable. However, this is not achieved by proving a simulation model is correct, since this is not possible, but rather to try and prove that the simulation model is in fact incorrect. If it is not possible to prove that a simulation model is incorrect the verification and validation process strengthens the credibility of the simulation model and its results [35]. It is thus essential that the verification and validation process be conducted during and after the development of a simulation model to ensure creditability. Complex real world problems can never be 100% accurately transformed into simulation models but the verification and validation aims to ensure that the model is sufficiently accurate [35].

4.3.1 Verification

The verification of a simulation model is to ensure that the real world system has accurately been transformed into a simulation model [35]. This is achieved by investigating and comparing all logical structures and parameters between the simulation model and physical scenario to confirm that the model accurately replicates the physical scenario [6].

According to Prof. Gunes [12] some common-sense suggestions to verify the simulation model is as follows:

1. Have someone else check the model.
2. Construct a flow diagram that includes each logical step of the model and explains every action performed by the simulation when events occur.
3. Closely examine the model output using a variety of different input parameter settings to confirm if they are reasonable.
4. Keep track of all initial parameters to ensure they are changed inadvertently through the course of the simulation.
5. Keep the model layout as clean and straightforward as possible.
6. Make use of a debugger.

7. If it is possible make use of a graphical representation to visualise the model.
8. If the operational model is animated, verify that what is seen in the animation imitates the actual system.

In the process of constructing the simulation model crude flow diagrams are used to assist with the flow of the model. These flow diagrams are refined and consolidated to in the end form Figure 4.2. As each new section of the simulation was coded and added to the simulation model multiple tests were performed to ensure nothing changed inadvertently through the course of the simulation. These tests included using functions that get flagged when parameters and variable change that should not. Further functions were coded that constantly monitor changes made to variables ensuring that the changes are in accordance to the assumptions and rules of the DC. If a function was triggered due to variable changing unexpectedly the entire simulation stops immediately and the variable that changed is highlighted. To ensure the more complex functions of the simulation model works as intended Python's built-in debugger (pdb) is used to step through code and ensure everything works correctly.

Visualising the simulation model does not substantially increase the amount of information gathered from the simulation. To visualise the entire model the best way is to display it in two dimensions. Doing this, however, removes the visualisation of the vertical movement. To prevent this the visualisation of the model can be done in three dimensions. If all the objects (storage racks, pallets, high lifts) are presented with 3D models they, however, block the movement of high lifts. This means that only small parts of the model can be visually inspected at a time. Instead of visualising the entire model small examples are visualised with animations to help explain the different movement logics, job sequencers and slotting algorithms. The code used in the simulation model is also used in the visual examples. Each example is visually and mathematically assessed to verify that they portray the actual system.

Using the test functions and visual examples the model it is verified that the simulation is accurately transformed into a simulation model. Finally the simulation model needs to be validated before it can be used.

4.3.2 Validation

Validation is a process to ensure that if the correct simulation model is used, the model is sufficiently accurate for the purpose of the simulation [35]. Validation is often applied to a simulation model to identify and rectify any anomalies present in the simulation model but not in the real world system to ensure the model replicates the real system as close as possible [6]. Validation is repeated multiple times on a simulation model to improve the model's accuracy until it is sufficient for its intended purpose.

Naylor and Finger [26] formulated a 3-step approach to the validation process of computer modelling that is widely used.

1. The simulation model should be built with high face validity.
2. Validate the simulation model's assumptions.
3. Compare the simulation model's input-output transformation to the corresponding input-output transformation of the real world system.

Similar to this is Robinson's [35] four processes of validating a simulation model.

1. Conceptual model validation: Determines if the scope and level of detail used in the simulation model is sufficient to reach the predefined objectives and that all assumptions are correct.
2. Data validation: Does the data used in the simulation accurately interpret the observed behaviour of the real world scenario.
3. White-box validation: Determining if each part of the simulation model accurately portrays the real world system. This can be done by
 - confirming that all aspects of the code work as intended.
 - doing an inspection to insure each component of the simulation model corresponds correctly with the component in the real world system.
 - Analysing the output data of individual components from the simulation model and comparing it to the expected outcomes.
4. Black-box validation: Comparing the overall results of the simulation model to the expected results of the real system.

Conceptual model validation inspects the level of detail of each component in the simulation model to determine if it contains the level of detail required to reach the proposed objectives accurately. As the proposed simulation model is agent based each entity, pallet and high lift comprise of unique information regarding their locations, ID's and time stamps. The high level of information for each entity implies that the simulation model contains a high level of detail. This indicates that the effect, of a pallet's location and the movement logic used, on the flow of the storage racks can accurately be portrayed.

Data validation determines if the data used in the simulation model is accurate. If the simulated data cannot be validated the results of the simulation bare no meaning [6]. Extensive analysis on the data is necessary for all the data used in the simulation model. Data analysis is performed in Chapter 3 and it is found that the data is accurate enough to use in the simulation model.

To ensure the simulation model accurately replicates real world scenario the simulation is continuously validated. The following conditions are monitored in the simulation model to confirm that a high face validity and white-box validation is achieved:

1. Each rack location can only support one pallet at a time. No pallet can thus be stored in a rack location that is already occupied.
3. High lifts are unable to remove pallets from rack locations unless the pallet is located in the specific location. This is inspected to ensure pallets are never teleported to a rack location.
4. All pallets are stored in the rack location assigned to them by the job list.
5. Confirm all assumptions are correctly implemented.
6. All pallets that are removed from the storage rack are placed in the racks before it is attempted to remove them.
7. High lifts can only move one pallet at a time.
8. Jobs are performed in the correct sequence as defined by the job list.
9. The correct number of jobs are completed.

10. High lifts never enter occupied aisles.

Although validating all the aspects that influenced the construction of the simulation it is not enough to prove the simulation's output bear any meaning. The output of the simulation model should statistically be proven to accurately portray the real world system.

4.3.3 Output validation

When validating the output of the simulation model it is crucial to repeat the simulation numerous times. By repeating the simulation the certainty that the output is accurate increase. The simulation can, however, not be repeated indefinitely. Burghout [4] formulated equation (4.1) to calculate the number of simulation recurrences required for a specific statistical confidence on the simulated output.

$$N(m) = \left(\frac{S(m)t_{m-1,1-\alpha/2}}{\bar{X}(m)\epsilon} \right)^2 \quad (4.1)$$

In equation (4.1) $N(m)$ denotes the number of times the model must be simulated. $\bar{X}(m)$ is defined as the real mean of the results from m simulation runs. $S(m)$ is the standard deviation of the results after m simulation runs. Further ϵ is the percentage error of the simulated mean defined as

$$\epsilon = (|\bar{X}(m) - \mu|)/\mu,$$

where μ represents the historical travel time. Lastly $t_{m-1,1-\alpha/2}$ is the critical value of the two tailed t-distribution with $m - 1$ degrees of freedom and a significance level α .

The travel times are validated as their accuracy are vital when comparing the different logics and algorithms. The first step in validating the travel times are to confirm that the method used to estimate travel times is accurate. Rack locations 1005, 2018, 2025, 3012, 3020, 4028, 5005 and 6016 are selected to validate the estimated travel times. The travel times, in seconds, for both storage and retrieval jobs to and from the selected rack locations are simulated. To properly test the accuracy the same number of travel times are simulated as data points available for each rack location.

Using equation (4.1) the number of simulations are calculated for each rack location for both storage and retrieval job types. The simulation is repeated 10 times and it can be said with 95% certainty that the results are sufficiently close to the actual completion time in seconds. Table 4.3 summarises the median historical and average simulated travel times to each of the selected rack locations. This comparison shows that by simulating the travel times, in seconds, for each rack location the real world travel times are accurately replicated.

Further the distributions of the simulated travel times, in seconds, are compared to that of the real world system. Using Levene's test the null hypothesis is tested that the simulated travel times and the historical travel times have equal variances. The alternative hypothesis is that the simulated travel times and the historical travel times do not have equal variances. Testing the null hypothesis with a statistical significance of $\alpha = 0.05$ it is shown in Table 4.4 that the null hypothesis cannot be rejected. This proves that the simulated travel times accurately replicates the historical travel times.

The simulated travel times is further validated by comparing the historical travel time distribution visually to the distribution of the simulated times. Referring to retrieval times for rack

Pallet information		Storage times (seconds)			Retrieval times (seconds)		
Rack location	Height	Historical	Simulated	Diff%	Historical	Simulated	Diff%
5	1	45.12	45.16	0.088%	44.26	44.89	1.413%
18	2	60.81	60.92	0.181%	44.89	45.01	0.267%
25	2	55.61	55.67	0.107%	48.79	48.78	0.021%
12	3	67.68	67.86	0.266%	42.31	42.15	0.379%
20	3	62.96	62.92	0.066%	46.63	46.45	0.387%
28	4	65.29	65.32	0.046%	52.99	52.60	0.739%
5	5	83.12	83.41	0.348%	47.35	47.45	0.211%
16	6	83.54	83.52	0.024%	55.19	55.03	0.290%

Table 4.3: A table summarising of a comparison between the average historical travel times and the estimated travel times.

Pallet information		Storage Distributions		Retrieval Distributions	
Rack location	Height	p-value	Rejected	p-value	Rejected
5	1	0.8801	No	0.7937	No
18	2	0.7434	No	0.8069	No
25	2	0.8713	No	0.7817	No
12	3	0.8023	No	0.7130	No
20	3	0.8679	No	0.7008	No
28	4	0.8595	No	0.7704	No
5	5	0.8411	No	0.8953	No
16	6	0.8817	No	0.7165	No

Table 4.4: Using Levene's test the null hypothesis is tested whether or not the simulated and historical travel times' distributions are the same.

location 3012 Figure 4.3 illustrates the comparison between the historical travel times and the simulated travel times in seconds. Figure 4.3 shows very little variation between the historical and estimated travel times. This further confirms the estimated travel times accurately represent those of the real world scenario.

To validate the travel distances, in rack locations, storage rack locations 1005, 2018, 2025, 3012, 3020, 4028, 5005 and 6016 are selected. Although the travel distances, in rack locations, are deterministic and not simulated they are calculated during the simulation. It should thus be validated that the model calculates the travel distance, in rack locations, correctly. The distances, in rack locations, are calculated for storage and retrieval jobs to each of these rack locations. Using the simulation model jobs are simulated to be stored and retrieved from these locations. The travel distances, in rack locations, calculated in the simulation model are compared to distances manually calculated using the historical data. Table 4.5 shows that the historical and calculated travel distances are equal validating that the simulation model accurately calculates the distance in rack locations.

Lastly, in the simulation model the travel times, in seconds, are validated by comparing it to the historic job list. For validation purposes only one 12 hour shift is simulated. The night shift of 14-Nov-2015 is used as the job list and consists of 158 jobs. Replicating the simulation 20 times the standard deviation ($S(m)$) of the simulated travel times is calculated as 92.3 seconds. The standard deviation is 0.83% of the simulated mean ($\bar{X}(m)$) of the total completion time, 11 118 seconds. From the historical data the actual completion time (μ) is found to be 11 181 seconds making $\epsilon = 0.6\%$. Substituting these values into equation (4.1) with a t-distribution critical value of 2.086 ($\alpha = 0.05$) it is calculated that one simulation is required to provide a solution that is sufficiently close to the actual completion time.

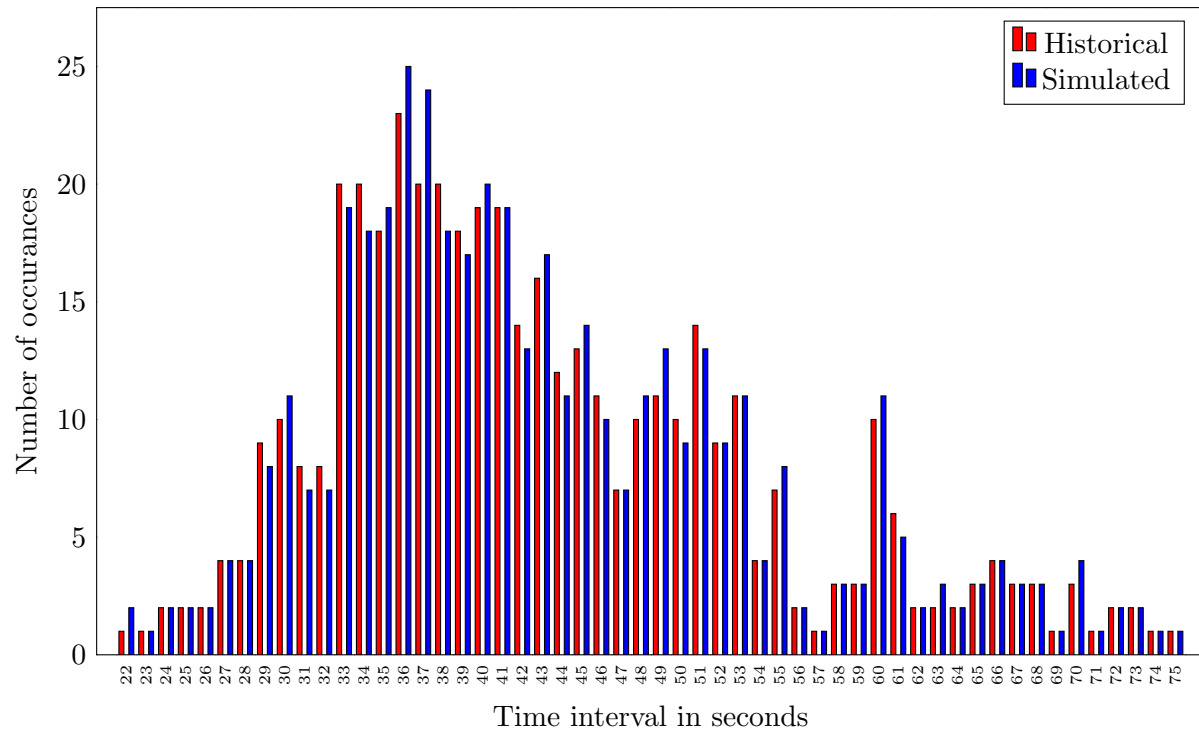


Figure 4.3: Graphical comparison between historical travel times and estimated travel times for the retrieval jobs performed on rack location 3012.

Pallet information		Storage distance (rack locations)		Retrieval Distance (rack locations)	
Rack location	Height	Historical	Simulated	Historical	Simulated
5	1	58	58	67	67
18	2	34	34	69	69
25	2	20	20	69	69
12	3	48	48	71	71
20	3	32	32	71	71
28	4	18	18	73	73
5	5	66	66	75	75
16	6	46	46	77	77

Table 4.5: A table summarising of a comparison between the historical travel distance and the calculated travel distances.

To gather more information the job list is simulated 150 times and it is seen that the average simulate travel time converges after 120 simulations, shown in Figure 4.4, after this point it stabilises at 11 028 seconds. The converged average simulated travel time, in seconds, differs from the historical observed travel time with 1.38% and falls within the selected significance level. Comparing the highest (11 428s) and lowest (11 024s) simulated completion times, in seconds, they simulated times differ by 2.2% and 1.4% respectively.

The sequence in which jobs are released and completed is logged by the simulation and compared to the historical job list. By comparing the simulated and historical job lists it is confirmed that jobs are completed in the correct order and that the correct number of jobs are completed.

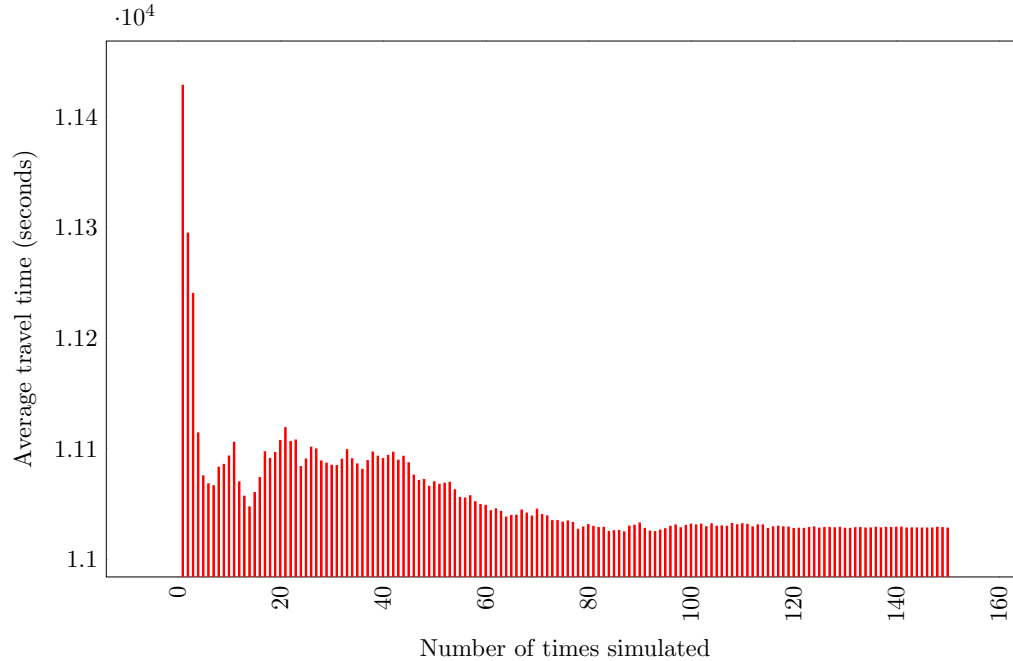


Figure 4.4: Graphical representation determining the number of simulations are needed before the results converge to a static value.

Unfortunately only the times spent carrying pallets can be determined from the historical data as the dead heading times are not recorded by the WMS. Although there is no way of validating the dead heading travel times and distances, it is calculated using the same mathematical formulas and logic as when a pallet is moved. Due to both being calculated in the same manner it is assumed that the dead heading distances and times are also correctly calculated.

4.4 Implementation

The validation and verification confirms the simulation model is capable of accurately simulating the real world scenario. Using the simulation model the effect of using different combinations of the proposed movement logics, placement algorithms and sequencer can accurately be presented. Having a validated and verified simulation model means that any job list, with the correct format, can be imported and would be simulated accurately. This allow for simulations to be conducted not only using PEP's historical data but job lists generated according to any specifications, within the scope of the simulation model, as well. The simulation model is combined with a decision support tool to increase user friendliness and broaden scope of users. The decision support tool is described in Appendix C.

CHAPTER 5

Simulation results

Utilising the simulation model described in Chapter 4 the proposed high lift movement logics, sequencing algorithm and placement algorithms are simulated. Before these logics and algorithms are implemented into a real world scenario an analyses of each is done. Allowing for a proper understanding of each logic and algorithm and each performs in different scenarios.

5.1 Sensitivity analysis

Using the simulation the effect of different parameters and scenarios for each logic and algorithm can greatly increase our understanding of each. For examination purposes the logics and algorithms are tested on a storage rack system as illustrated in Figure 5.1 with dimensions $m = 6$ and $n = 32$. All simulations are performed using a computer with a 3.9GHz processor and 8GB RAM.

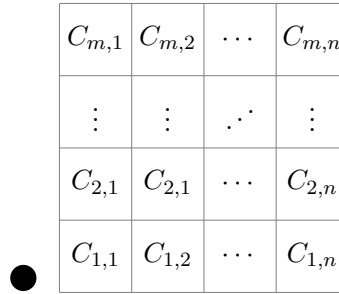


Figure 5.1: A schematic representation of the storage rack system with the order picking area being represented by the black circle.

To fully grasp the possibilities of a logic or algorithm job lists, each consisting of 50 jobs, are used. The jobs are randomly generated with the same data format depicted in Table 5.1. From the headings in Table 5.1 the **Job type** indicates if a pallet is stored or removed from the storage racks for a specific job. The **Rack loc** indicates where in the rack system the pallet is stored or removed from. **Pallet ID** is used to track the pallet on the storage rack system. The **RC value** indicates how long a pallet will remain in the storage rack system. An RC value of 1 indicates the pallet is stored for less than 7 days with each RC value increasing the storage time in increments of 7 days. **Start loc** is the queueing location for storage pallets. **End loc** is the queueing location for all retrieval pallets.

#	Job type	Rack loc	Pallet ID	RC value	Start loc	End loc
1	IN	AA03004	99735	3	33	0
2	IN	AA05010	89546	2	33	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
24	IN	AA04014	97485	3	33	0
25	IN	AA03028	64701	4	33	0
26	OUT	AA04011	32028	3	33	0
27	OUT	AA03018	87546	1	33	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
49	OUT	AA03030	28704	2	33	0
50	OUT	AA03022	52196	3	33	0

Table 5.1: Table representing the general format of a job list. In this case a job list consisting of more inbound jobs (12) than outbound jobs (8) are recorded. The **Pallet** ID makes tracking a pallet effortless. **Start loc** and **End loc** are the queues in which inbound and outbound pallets are placed respectively. These queue locations are situated at the ends of the storage rack system giving them a number of $m + 1$ or $m = 0$ depending on the high lift logic used.

Because the job lists are randomly generated there is no historical data that can be used for comparison. This means equation (4.1) cannot determine the number of simulation replications needed for the results of these job lists to be deemed accurate. Kelton *et al.* [14] formulates equation (5.1) that is capable of calculating the required replications without the need for historical data.

$$N = n_0 \times \left(\frac{t_{m-1, 1-\alpha/2} \times s}{h\sqrt{n_0}} \right)^2, \quad (5.1)$$

where,

n_0 is the size of the population,

s is the standard deviation of the population,

h is allowed the error of the mean and

$t_{n_0-1, 1-\alpha/2}$ is the two tailed t-distribution critical value with $n_0 - 1$ degrees of freedom with significance level α .

Using equation (5.1) the required replications are calculated with a significance level (α) of 0.05. Further more enough replications need to be performed to have a 95% certainty that the after N replications the mean is within 1% of the actual mean. The initial number of runs (n_0) is selected as 10 for accuracy and improved run time.

5.1.1 Movement logics analysis

An analysis is required to get a better understanding of how each of the five high lift movement logics, as described in § 2.1, compare to each other. This is achieved by comparing the effect each movement logic has on the total movement distance, in rack locations and the completion time, in seconds, when using job lists of different configurations. The composition of storage and retrieval jobs can rarely be controlled in the real world. This means there is not always an equal number of storage and retrieval jobs that are executed per job list. For this reason job lists are generated with one of three possible configurations namely, **Storage** > **Retrieval**, **Storage** < **Retrieval** and **Storage** = **Retrieval**. A **Storage** > **Retrieval** indicates that

the job list consists of $\pm 75\%$ storage jobs. A $\text{Storage} < \text{Retrieval}$ job list indicates the job list consists of $\pm 75\%$ retrieval jobs. A $\text{Storage} = \text{Retrieval}$ job list consists of an equal number of storage and retrieval jobs.

Movement distance analysis

Due to the movement distance being deterministic, simulating a single job list multiple times would not change the outcome of the total movement distance. The horizontal and vertical movement distances can, however, give insight to the expected completion time. Depending on a high lift's horizontal and vertical movement speeds the completion time of the same job can change drastically. Investigating the effect different high lift movement speeds have on a job list helps with identifying what type of high lift is best suitable for which of the different movement logics, sequencing algorithms and placement techniques.

Job lists are filled with 50 uniformly generated jobs for each configuration. The number of job lists that are generated, for each combination, is determined using equation (5.1). The total movement distance, in rack location, are calculated for each job list. The average movement distance to complete a job list for each movement logic is plotted in Figure 5.2.

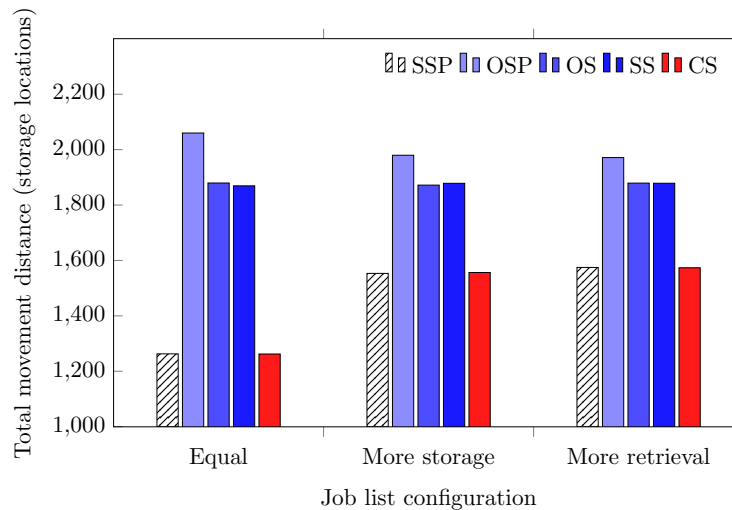


Figure 5.2: Plots of the average cumulative movement distance to complete a job list for each configuration type.

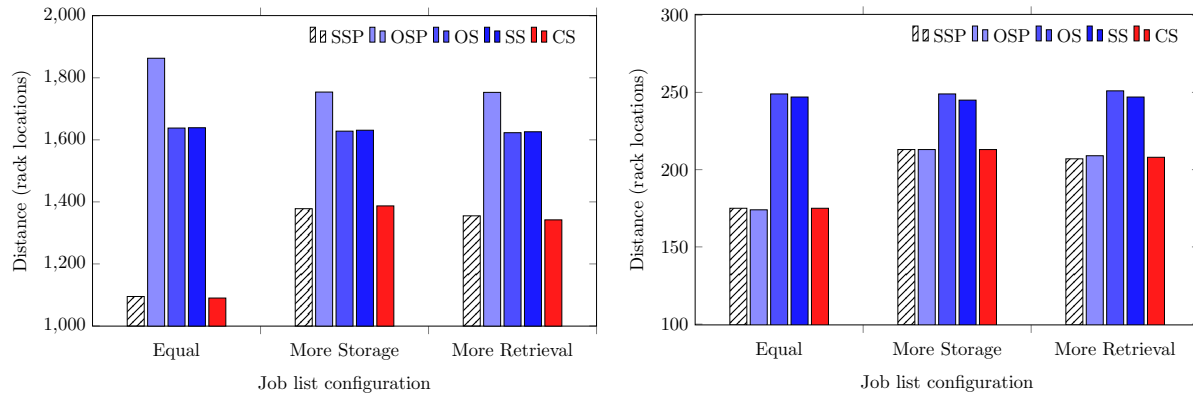
Comparing the configuration only the high lift movement logics with pairing logics are significantly different. Figure 5.2 shows that the SSP and CS movement logics perform the best in all tested configuration. Irrespective of the movement type configuration non-pairing movement logics perform similar with less than 1% difference between them. The change in average distance per job is only fractions of a rack location's width. This means that when using non-paired logics the composition of a job list does not affect the total movement distance.

The SSP and CS movement logics perform better due to the job pairings that are made. Pairing jobs reduces the dead-heading endured for each job resulting in the total movement distance decreasing. As the ratio between storage and retrieval jobs increase these movement types can form less job pairings. If no more job pairing can be performed pairing movement logics change to non-pairing movement logics. This means that as the number of jobs pairing decreases the closer the pairing movement logics start to resemble non-pairing movement logics. Although the OSP movement logic also make use of pairing its total movement distance is the highest. This

is due to the OSP logic's queueing locations that forces an entire aisle length of dead heading after each job pairing is completed. Switching to a non-pairing movement logic allows the OSP logic to exclude the aisle length of dead heading making the total distance shorter.

The performance of each movement logic is tested using high lifts with different horizontal and vertical movement speeds. This is achieved by splitting the total movement distances into their vertical and horizontal components, Figure 5.3.

Investigating Figure 5.3(a) shows that it resembles the shape of Figure 5.2 very closely, except for the OSP movement logic. The difference percentage between the movement logics for each configuration it differs with less than 1% between the two figures (Table D.1 and Table D.2). Figure 5.3(b) shows that the pairing movement logics have very similar vertical movement distances, less than 1% difference (Table D.3), for all configurations. The same is true for the non-pairing movement logics (Table D.3). This means that the erratic increase in movement distance observed for the OSP movement logic in Figure 5.3(a) is caused by the horizontal dead-heading endured while moving the entire aisle from one queue to the other.



(a) Average total horizontal movement distance of the total completion distance.

(b) Average total vertical movement distance of the total completion distance.

Figure 5.3: Graphical breakdown of the average horizontal and vertical total movement distance, in rack locations, for each movement logic and configuration.

From Figure 5.3 it can be seen that high lifts move a far greater distance horizontally than vertically. Utilising the tested storage rack layout about 85% of the total movement distance comprises of horizontal movement, Table D.4. This is expected due to the dimensions of the tested storage rack system – 32 rack locations in width and 6 levels high. This means that for this system high lifts capable of moving faster horizontally would pose the greatest benefit.

Completion times analysis

Job lists are generated for each configuration - the number of job lists are determined using equation (5.1). Each job list consists of 50 uniform randomly generated jobs. Pallet rack locations (**Rack loc**) and RC values (**RC value**) are given values at random. Applying each high lift moving logic to the different job lists the average completion times, in seconds, are simulated using the travel times discussed in Chapter 4. The average simulated completion times, in seconds, are plotted in Figure 5.4.

For each movement type configuration the best performing movement logic is used as the base line performance. The average completion time, in seconds, for each movement logic is then compared to this base line for each configurations. The difference between movement logics

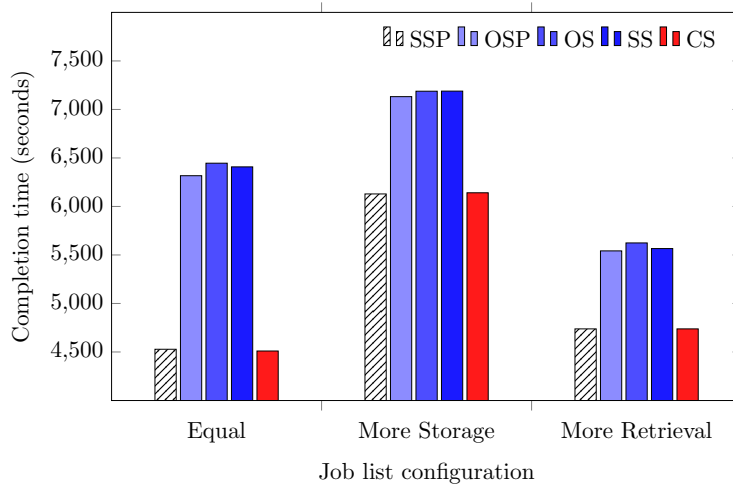


Figure 5.4: Plot of the average completion time of job lists different configuration types.

are measured in terms of percentages and is summarised in Table 5.2. The best performing movement logic for each movement type configuration is indicated by “—”.

	SSP	OSP	OS	SS	CS
Storage = Retrieval	—	40.07%	42.93%	42.08%	—
Storage > Retrieval	—	16.36%	17.28%	17.29%	—
Storage < Retrieval	—	16.97%	18.70%	17.48%	—

Table 5.2: A summary of the difference percentages between the average completion times, in seconds, of the best movement logics compared to the other movement logics.

Investigating Figure 5.4 and Table 5.2 shows that the difference between the SSP and CS movement logics are negligently small ($<0.5\%$) and performs equally well for all configurations. Furthermore, it shows that irrespective of the configuration the SSP and OS movement logics perform the best, for that configuration. This is not the case for the other three movement logics and they perform the best when more retrieval jobs are performed.

Inspecting Figure 5.4 it shows that as the ratio of storage jobs to retrieval jobs increase so does the completion time for non-pairing movement logics. The increase in completion time is explained by the difference in travel times between storage and retrieval times. Based on the historical data it takes a high lift longer to complete an storage job than a retrieval job.

Table 5.3 summarises the average completion time, in seconds, for job lists only consisting of storage or retrieval jobs. It takes on average 62.4% longer for the job lists that only consisting of storage jobs to be completed than when it only consists of retrieval jobs. Furthermore, Table D.6 shows that difference percentage between each movement logic doing only storage or retrieval jobs is negligibly small ($\leq 1\%$). This indicates that irrespective of the movement logic used if a job list only consists of one movement type they all perform the same.

Figure 5.4 shows the higher travel times endured when storing pallets also causes the total completion time to increase for the movement logics that pair jobs. This, however, is only true when there is an unequal number of storage and retrieval jobs.

The job pairing movement logics are able to reduce the total completion time because of the job pairings they form. The pairing movement logics are formulated to revert to non-pairing movement logics when no more job pairings can be formed. This causes the total completion time to increase when no more job pairing can be formed. The increase in completion time

	SSP	OSP	OS	SS	CS
Only retrieval jobs	4920	4906	4906	4896	4928
Only storage jobs	7994	7963	7984	7975	7972
Time increase	62.5%	62.3%	62.7%	62.8%	61.8%

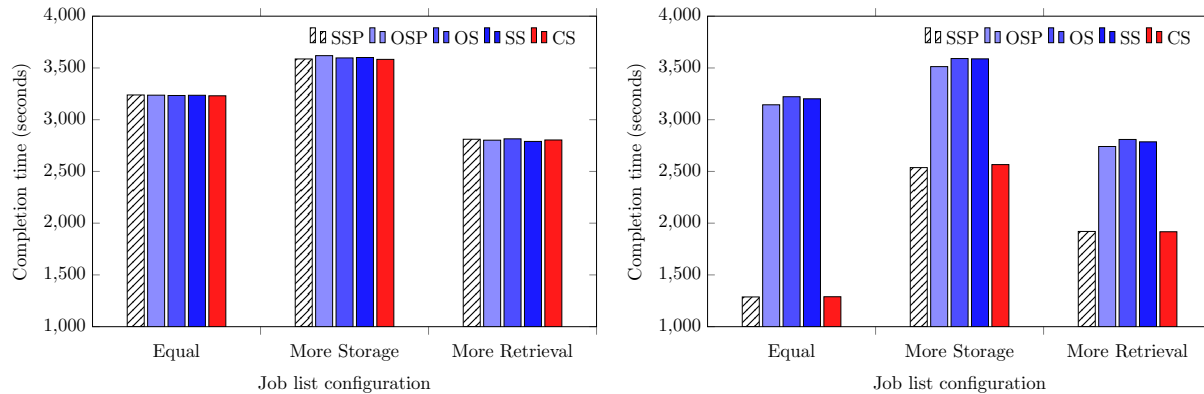
Table 5.3: A summary of the average total distance, in rack locations, each movement logic took to complete a job list of specific configuration.

is governed by which movement types remain. All the remaining jobs must be of the same movement type, either storage or retrieval. If this is not the case more job pairings can be formed. In the case that there are storage jobs remaining the total completion time is increased the most, as seen with the non-pairing movement logics.

The SSP and CS movement logics perform the best when there is an equal number of storage and retrieval jobs. This is due to these movement logics being able to pair all the jobs in the job allowing for the most reduction in completion time.

Utilising the OSP movement logic with different configurations is, however, not effected in the same manner as the other two paired movement logics. Figure 5.4 shows that the total completion time, when using the OSP movement logic, increases as the number of storage jobs increases - similar to the non-pairing movement logics.

Further investigation is needed to determine why the OSP movement logic does not perform its best when a job list consists of an equal number of storage and retrieval jobs. To better understand what causes the higher completion time for the OSP movement logic the dead-heading and occupied components of the completion times are investigated.



(a) Average occupied time components of the total completion time.

(b) Average dead-heading time components of the total completion time.

Figure 5.5: Graphical breakdown of the average dead-heading and occupied completion times, in seconds, for each movement logic and configuration.

Comparing the dead-heading component, Figure 5.5(a), and the occupied component, Figure 5.5(b), leads to a better insight into what causes the increased completion times. The comparison shows that irrespective of the movement logic being used the average total occupied time, in seconds, is very similar. Referencing Table D.5 it shows that this is indeed the case. All increases between the best performing movement logic and the worst are less than 1% for each job list configuration.

The same does, however, not hold when inspecting the average total dead-heading times, in seconds. The dead-heading times are more in-line with what is seen with the average completion

times. This is expected since the occupied times are similar for all movement logics for the different job list configurations. From the comparison it is clear that the OSP movement logic does not perform as good as the other two job pairing movement logics due to its increased dead-heading time.

It is believed that the increase in the dead-heading times are caused by the OSP logic's queueing locations. The OSP movement logic is forced to perform an entire aisle length of dead heading after each job pairing is completed. Figure 5.6 compares the three job pairing movement logics if the aisle length dead-heading time is ignored. This shows that if the OSP movement logic was capable of not dead-heading the entire aisle after each job pairing all three movement logics would perform equally well for all movement type configurations. This also indicates how dependent the performance of the OSP movement logic is on the speed at which a high lift can move from one queue to the other.

To improve the OSP movement logic the effect that the aisle movement has on the dead-heading time needs to be decreased. The easiest way to achieve this is by using the dead-heading time for more productive means. From this the OS movement logic transpired by instead of dead-heading the aisle the high lift performs another job pairing.

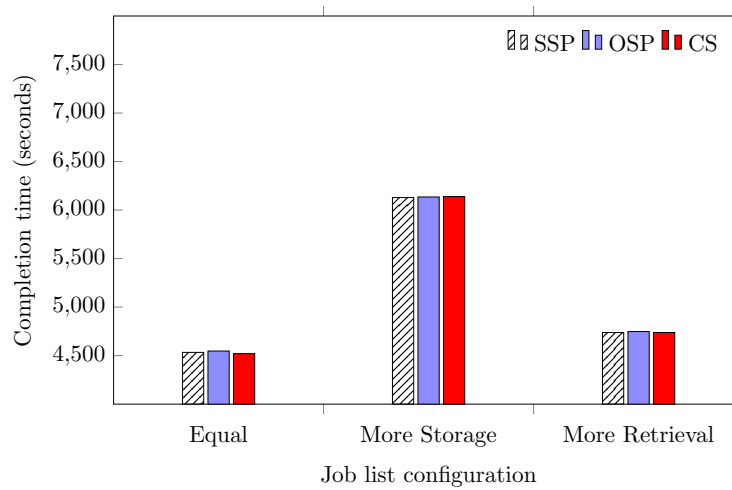
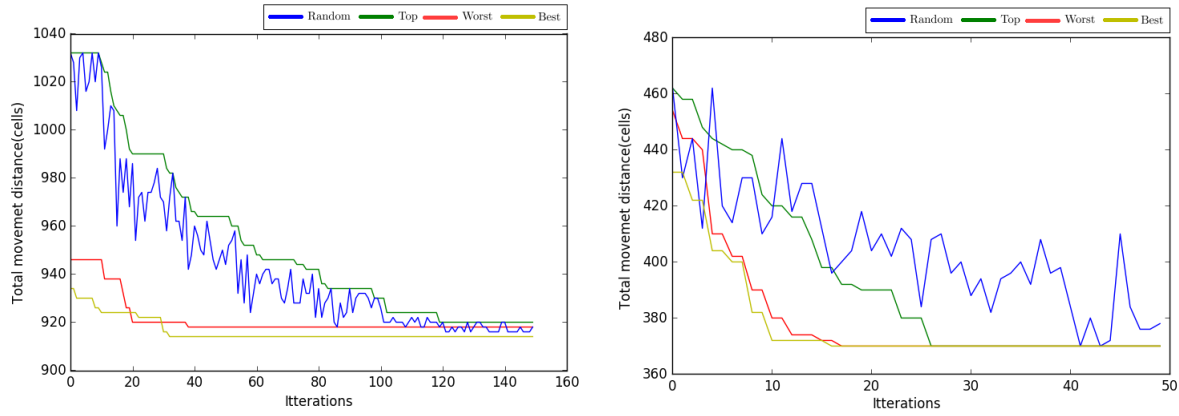


Figure 5.6: Plot of the average completion time, in seconds, of job lists for the different configuration types if the aisle length dead-heading time is ignored.

5.1.2 Sequencing algorithm analysis

In a further attempt to improve the total average completion time, in seconds, and movement distance, in rack locations, the sequence in which jobs are completed is inspected. Comparing the time needed to obtain the optimal job list sequence for the five different sequencing algorithms, the best is selected. Changing the sequence of the job list only significantly effects movement logics that make use of pairing jobs. For this reason only job lists with an equal ratio of storage and retrieval jobs are considered when comparing the sequencing algorithms as this configuration give the most information regarding the movement logic.

Figure 5.7 shows the number of iterations needed for each Tabu-search algorithm to converge to a local minimum. The yellow and red algorithms are seen to converge in the least number of iterations. All four algorithms, however, converges to different distances. When trying to determine a good sequence of certain job lists the algorithm forms cycles at local minimums. Whether or not cycles are formed is dependent on the initial order of the job list. According to



(a) Sequencing algorithms getting stuck in local minimums.

(b) Sequencing algorithms converging to same minimums with the use of randomness function.

Figure 5.7: Graphical comparison of how adding a randomness to the sequencing algorithms. Figure 5.7(a) shows that with certain job lists not all the movement logics converge to a single value. Figure 5.7(b) shows that when using a randomness function in each sequencing algorithm they all, except the random sequencer, converge to the same value.

Edmund Burke [5] adding randomness to the algorithm can act as a good anti-cycle mechanism, also known as diversification. In an attempt to prevent cycles forming a randomness modifier is added to reshuffle the job list after a certain number of iterations has passed with the solution staying the same. It is found that the best results are obtained when the modifier is activated after the solution has not changed for 10 iterations. Although shuffling the job list increases the total movement distance it gives the Tabu-search possible alternative routes to find a better sequence. Figure 5.7(b) shows that when the randomness modifier is applied to the algorithm, each algorithm (except the random tabu search) converges to the best known solution.

By sequencing storage and retrieval jobs correctly for paired job movement logics it is possible to exclude unnecessary back tracking within the storage racks. This is explained by looking at a simple example applying the OSP movement logic to a storage rack system depicted in Figure 5.8. First a bad sequence is considered with the first job pairing being the storage job destined for rack location labelled 4 and the retrieval job in rack location labelled 6. The second job pairing consists of the storage job destined for rack location labelled 7 and the retrieval job in rack location labelled 2. Using this sequence forces the high lift to back track to the rack location 6 after the storage pallet is placed in rack location 4. The high lift back tracks the length of two rack locations and gives a total travel distance of 44 cells to complete the job list.

Improving the job list sequence a pairing is formed between the storage job destined for rack location labelled 7 and the retrieval job in rack location labelled 6. The next pairing is the storage job destined for rack location labelled 4 and the retrieval job at rack location labelled 2. This sequence does not need any back tracking and requires $2(n + 1)$ cells to complete a pairing giving a total travel distance of 40 cells.

This means that to obtain a good sequence for a job list, jobs must be paired in such a way that pairings have the storage job further away from the retrieval queue than the retrieval job. This method of reducing the back tracking distance applies to all the different movement logics. The CS movement logic is handled in the same way as in the example for the OSP movement logic expect the total distance is $n + 1$ for each pairing. The SSP movement logic with a good job list sequence essentially excludes the retrieval jobs when calculating the total movement distance as it is already on the high lift's path when it moves back to the queue after storing a pallet.



Figure 5.8: A schematic representation a paired jobs with opposite side queueing movement logic. The black blocks represents retrieval pallets and the brown blocks represents storage pallets. IN and OUT refers to the retrieval and storage queues receptively. The numbers represents the label for each storing location.

Applying the SSP movement logic to the example portrayed in Figure 5.8 the job list is sequenced first jobs pairing consists by pairing the storage job in rack location labelled 4 and the retrieval job in rack location labelled 6. The second job pairing is then between the storage job in rack location labelled 7 and the retrieval job in rack location labelled 2. Using this sequence the high lift moves further down the aisle to reach the retrieval pallet in rack location 6 resulting in extra travel distance. A total movement distance of 26 rack locations is endured when using this bad sequence.

The sequences is improved by rather pairing the storage job in rack location labelled 4 and the retrieval job in rack location labelled 2. The next job pairing is then between the storage job in rack location labelled 7 and the retrieval job in rack location labelled 6. By improving the job list sequence the retrieval pallets are always situated closer to the retrieval queue than its paired storage job. This results in only the movement distance endured when completing the storage jobs counting. By using the goof sequence the total travel distance is reduced to 22 rack locations.

Table 5.4 shows the average computational time required to find the best known sequence for each movement logic. The computational times are calculated using 1000 randomly generated job lists. Each job list consist of 25 storage jobs and 25 retrieval jobs uniformly spread over the storage rack system. Although the computational times for the Tabu-search algorithms are high it is unlikely that a single job list would have this many of jobs in real world applications.

	SSP	OSP	OS	SS	CS
Random Pairing	16.8	16.3	15.6	15.4	16.7
Pairing from top	17.8	17.2	16.9	17.2	18.0
Pairing from worst	14.9	14.5	14.3	14.1	15.1
Pairing from best	16.8	16.9	16.3	16.4	16.8
Hungarian method	0.02	0.03	0.02	0.02	—

Table 5.4: A summary of the average time, in seconds, each sequencing algorithm takes to converge to the best know solution.

Table 5.4 show that the Hungarian algorithm out performs all the other algorithms and is thus used to determine the best sequence. The Hungarian algorithm is, however, unable to determined the best job list sequence for the CS movement logic. The CS movement logic changes queueing locations after each job pairing is completed. This makes it hard to be implemented into the Hungarian algorithm and thus the **Pairing from worst** Tabu-search algorithm is used to sequence the job lists when this movement logic is used.

The effect of using the best known sequence is investigated by comparing it to a randomly sequenced job list, as done in § 5.1.1. Equation (5.1) determines the number of job lists that are simulated for each movement logic. Only job list with an equal number of each movement type

is considered as this highlights the effect of proper sequencing the most. Each job list consists of 50 uniform randomly generated jobs that is then properly sequenced using the Hungarian method. The results of simulating a best know job list sequence is plotted in Figure 5.9.

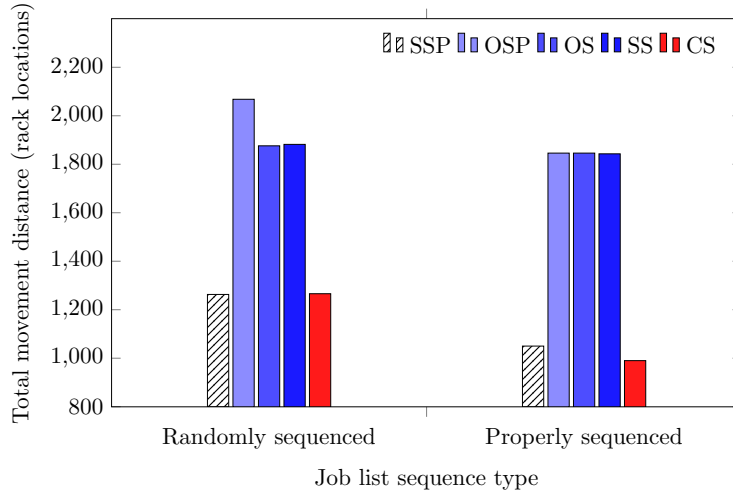


Figure 5.9: Plots of the average cumulative movement distance to complete a job list that is properly sequenced against a randomly sequenced job list.

Investigating Figure 5.9 shows that the movement logics that do not make use of pairing is not significantly affected by the sequencing. By using the sequencer together with the SS and OS movement logics the total movement distance is only decreased by 0.25% and 0.85% respectively. This is expected as the sequencer algorithm aims to improve the total movement distance by forming good job pairings. The non-pairing movement logics only have one job pairing – when high lifts transition from performing storage jobs to retrieval jobs.

The effect of using a best sequence for a job list on the total movement distance is much more apparent for the job pairing movement logics. Applying a sequencer algorithm together with the SSP, OSP and CS movement logics decreases the total movement distance by 17.27%, 10.68%, 21.95% respectively. An interesting observation is that the OSP movement logic, that is on average the worst performing, performs better than the non-pairing movement logics when a sequencing algorithm is applied. Furthermore, when a random sequence is used the SSP and CS movement logics perform identically, based on average total movement distance, but when a sequencing algorithm is introduced the CS movement logic perform 5.65% better than the SSP movement logic.

Similarly the average total completion times, in seconds, are investigated. Figure 5.10 summarises the comparison between a randomly sequenced job list and a job list that has a good sequence. The improvement in the average completion time is not as significant as the improvement in distance. The completion time improvement for each movement logic is 4.14%, 0.57%, 0.20%, 0.22% and 3.94% respectively.

Applying the sequencer algorithm reduces the dead-heading distance and travel time a high lift moves from the storage job's rack location to the retrieval rack location. This means that the improvement in time is dependant on how fast a high lift is capable of moving the distance between the rack locations. Based on the historical data the high lifts utilised in PEP's DC is capable of moving $\pm 1.43 \text{ m.s}^{-1}$. This means that an improvement of only a few seconds is achieved after each well sequenced job pairing. The improvement in completion time will, however, increase as the speed a high lift is capable of moving increases.

By sequencing jobs properly the improvement in completion time and distance is dependent on

the storage location of pallets. Even if jobs are properly sequenced the improvement can be insignificant if pallets are stored and removed from rack locations not suitable for the current movement logic. It is thus important to understand and investigate the effect changing the rack location of storage jobs have on the sequencer algorithm as well as on the different movement logics.

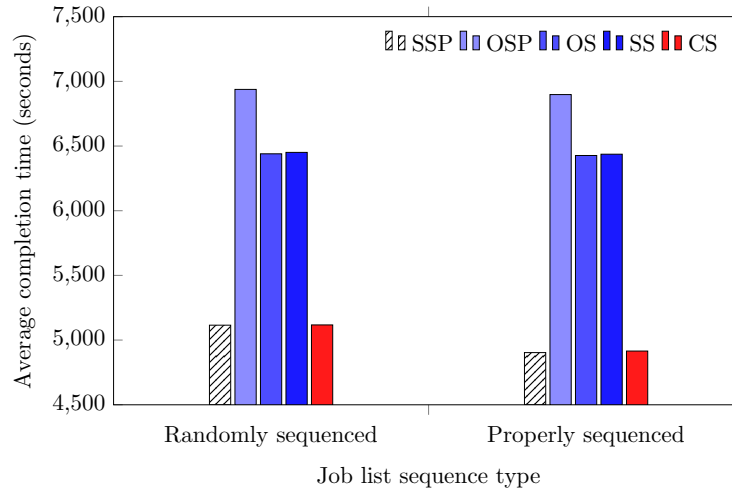


Figure 5.10: Plots of the average cumulative completion time, in seconds, to complete a job list that is properly sequenced against a randomly sequenced job list.

5.1.3 Slotting algorithm analysis

The five pallet allocation configurations are analysed in a final attempt to decrease the movement distance. Using the configurations described in § 2.3.2 the effect that each configuration has on the five different movement logics are investigated. A major issue arises when slotting according to a fixed criteria. By defining certain rack locations to only accommodate pallets with specific RC values the storage rack often runs out of storage space for pallets for certain RC values. It is thus important to determine which slotting configurations allow for the best usage of storage space and allows for the most pallets to be stored.

Irrespective of the high lift movement logic used the number of rack locations available for each RC value's slotting region stays constant. This means to test the rate at which the storage racks are filled only the different slotting configurations need to be analysed. To make the process a bit simpler only storage jobs are considered and the initial storage rack is completely empty. Jobs are created by generating their storage locations and RC values uniformly. Jobs are created until the first instance occurs of a job, with a specific RC value, not being stored due to a shortage of available storage locations. Repeating the process 1000 times for each slotting configuration the results are summarised in Table 5.5.

Table 5.5 shows the number of jobs that where stored (**Pallets stored**) before the first pallet was unable to be stored for each slotting configuration. It is also shown how many empty storage locations remained (**Available location**) on average as well as the occupied percentage (**Stored(%)**) before a pallet was unable to be stored. The high available percentage when using the row slotting configuration is due to entire rows being reserved for specific RC values. Without altering the shape this configuration forces certain RC value regions to be bigger than others. In a scenario where pallets are assigned RC values uniformly a slotting configuration that does not make all RC value regions of equal size will be at a disadvantage.

	Pallets stored	Available location	Stored(%)	RC1	RC2	RC3	RC4
Row	118.39	73.61	61.66	64	64	32	32
Column	166.60	25.40	86.77	48	48	48	48
Step	150.16	41.84	78.21	48	48	48	48
XY	155.79	36.21	81.14	46	50	48	48
Pythagoras	152.02	39.98	79.18	51	48	48	45

Table 5.5: A summary of the average number of uniformly random generated jobs that are stored, for each slotting configuration, before there are no more storage locations available. Also summarised is the size of each RC value region, in storage locations, for the different slotting configurations

In order to utilise the entire storage rack the shared regions are introduced. These shared regions allow for the storage rack to be utilised at maximum capacity. Using the shared regions also allows for all arriving pallets to be stored

The five pallet allocation configurations are analysed in a final attempt to decrease the movement distance. Using the configurations described in § 2.3.2 the effect that each configuration has on the five different movement logics are investigated. The placement configurations are analysed using generated job lists that comprise of 25 storage and 25 retrieval jobs. These jobs are distributed uniformly over the an empty storage rack. The RC values for each job is generated randomly using a uniform distribution. The number of replication per movement logic for each is determined using equation (5.1) and the results are summarised in Table 5.6.

	Unslotted	Row	Column	Step	XY	Pythagoras
SSP	1266.7	1023.2	1156.5	1029.9	1157.4	1187.4
OSP	2059.6	1816.9	1954.1	1834.7	1954.2	1938.2
OS	1869.9	1263.7	1664.1	1407.9	1664.2	1607.3
SS	1881.1	1255.6	1629.6	1280.9	1632.3	1702.2
CS	1266.7	1016.7	1147.9	1026.8	1148.1	1164.9

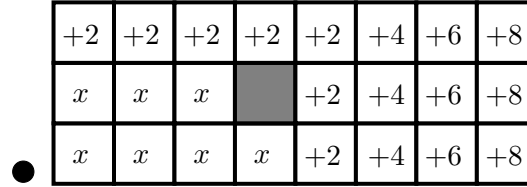
Table 5.6: A summary of the average total distance each movement logic took to complete all jobs with different slotting configurations.

Looking at the data summarised in Table 5.6 the row slotting configuration performs the best for all the high lift movement logics. Calculating the percentage difference between the unslotted and row slotted storage racks it is found to be 20.9%, 12.5%, 39.2%, 39.5% and 21.8% respectively. The greatest decrease is observed when using a high lift movement logic that does not utilise job pairing. Note that the step slotting configuration is able to get distances very close to that of the row slotting configuration. The step slotting configuration differs from the row slotting configuration by 0.6%, 1.0%, 10.8%, 2.0% and 1.0% respectively for each movement logic. This means that if the row slotting configuration is not a viable option the step slotting configuration may be used without increasing the distance substantially.

These result are obtained assuming the RC value of jobs are uniformly distributed. In practice it is not always the case meaning scenarios need to be tested with different ratios of RC values. Since the volume of pallets with specific RC values are DC specific there are to many combinations of RC value ratios to test making it impossible to formulate a general analysis. Instead a few observation are discussed.

Placing pallets at certain locations effect the total movement distance less than others. This is especially true when using a high lift movement logic that utilises the pairing of jobs. Figure 5.11

illustrates this point with the filled rack location representing the location of a retrieval job. When a job is stored in a rack location marked with a x it implies the distance to complete the job pairing is an arbitrary value x . For each row/column a storage job is placed further from the storage queue than the retrieval job the distance increase. As shown in Figure 5.11 each row/column adds an extra two rack locations to the total distance for each row/column the storage job is placed further.



+2	+2	+2	+2	+2	+4	+6	+8
x	x	x		+2	+4	+6	+8
x	x	x	x	+2	+4	+6	+8

Figure 5.11: A schematic representation of how the placement of storage jobs affect the the total distance in a system that utilises pairing of jobs. The darken rack location represents the retrieval job being paired with. x is defined as the minimum to complete the job pairing if the storage job is placed in any of these locations. The plus values represents the increase in distance, in rack locations, if the storage job is placed in it.

This shows the importance of looking at the locations of the retrieval jobs when determining the storage locations of pallets. When slotting pallets it is essential to store pallets in locations that potentially forms the best job pairings.

5.2 Historical data analysis (PEP)

The analysis performed on general instances for each movement logic and algorithms are applied to historical data received from PEP (Chapter 3). The job list described in § 3.6 is used, consisting of 10 consecutive shifts comprising of 4415 jobs, for simulation. These shifts are selected to ensure that an accurate and sensible comparison can be made between the different combinations of logics and algorithms. The total cumulative travel times and travel distances are calculated for each simulation and used as the primary comparison variable. Additionally the cumulative dead-heading time, dead-heading distance, jobs completed per hour and high lift efficiency is calculated for each simulation and used for a secondary comparison. The primary comparison determines which combination is the best suited for the tested job list as it represents the most important variables.

Investigating the job list further reveals information regarding what to expect from the simulations. Investigating the total number of storage jobs, 2481, and the total number of retrieval jobs, 1934, over all 10 shifts make it seem that there are ample job pairing possibilities. This is, however, not the case when inspecting each shift individually and even more so when looking at the aisles in each shift.

5.2.1 Job split

Table 5.7 shows that for most shifts there is a large difference between the number of storage and retrieval jobs. In some cases it does, however, show the difference to be very small as with shift 2 and shift 8. The small difference would indicate most jobs are able to be paired.

The jobs distribution of shift 2 is investigated for each storage rack as this shift almost comprises of a equal number of storage and retrieval jobs. Figure 5.12 illustrates the difference between the

Shift	Storage jobs	Retrieval jobs	Diff %
1	109	47	79.49
2	285	266	6.90
3	408	208	64.94
4	187	145	25.30
5	423	238	55.98
6	167	290	53.83
7	285	155	59.09
8	174	180	3.390
9	343	240	35.34
10	100	165	49.06
	2481	1934	24.78

Table 5.7: A summary of the total number of storage and retrieval jobs for each shift with their difference percentage.

number of storage and retrieval jobs per storage rack (1-46) for shift 2. The closer each storage rack's graphs, representing the storage and retrieval jobs, align the better the movement logics that utilises job pairing will be executed. Logics with job pairing changes to a single job logic when no more job pairings are able to form. It is thus expected that for some of the storage racks there is little to no difference between logics with job pairing and single job logics, with the same queueing configuration. The large difference in storage and retrieval jobs in each storage rack for the simulated job list means that the results may not correlate with the analysis of the movement logics discussed in § 5.1.

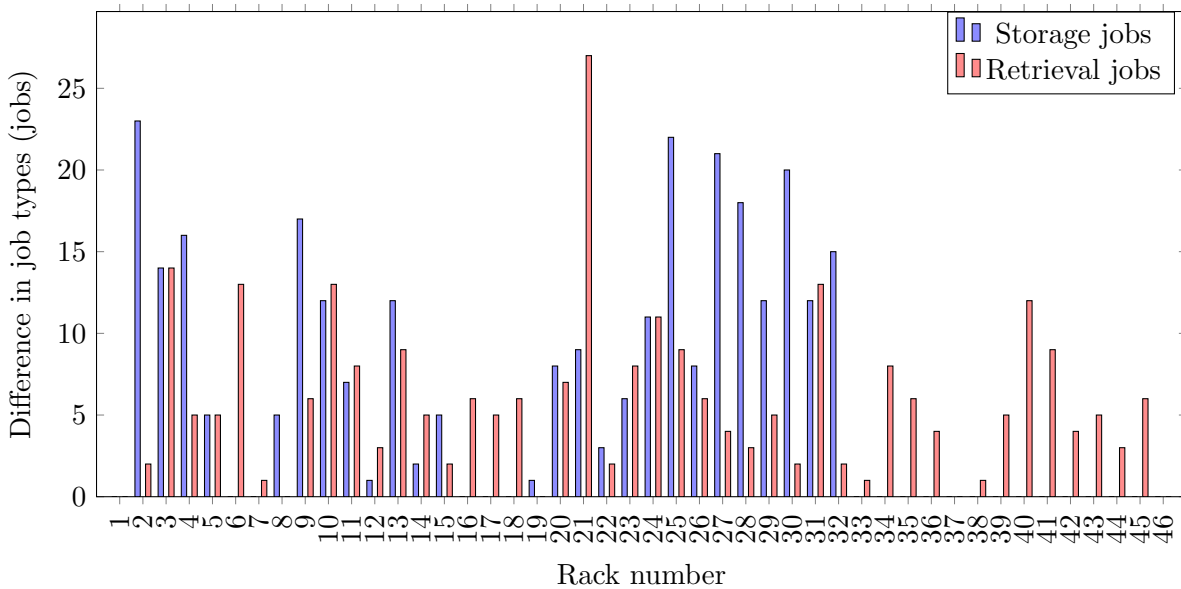


Figure 5.12: Graphical representation of the number of storage and retrieval jobs performed in each storage rack for shift 2. For each rack number the number of storage jobs is represented by the bar left of the rack number tick. Similarly the number of retrieval jobs is represented by the bar right of the rack number tick.

The distribution of pallet locations are further inspected for each storage rack. Each storage rack distribution is split into two categories – the **front**, closer to the order picking area, and the **back**, closer to the centre of the storage rack system. If a pallet's storage location is a length less than 19 rack locations from the order picking area it is classified as a **front** location. This value is selected as the furthest a pallet can be from order picking area is 37 rack locations, the

half way mark is thus 19 rounded up. All storage locations further than 19 rack locations from the queue closest to the order picking area is classified as **back** locations.

Table 5.8 shows a comparison between the number of storage racks that have a majority of rack locations categorised as **front** against those categorised as **back**. In the case that the number of **front** and **back** storage racks do not add up to 46 it means that not all storage racks were used for the specific shift. It is shown that out of the 10 shifts only two shifts have storage racks that are categorised as being **back** configurations. This means the job list has a strong tendency to have pallets be removed and stored closer to the order picking area. Over the span of the 10 shifts a total of 209 storage racks are being categorised as **front** while 148 are classified as **back**. As discussed in § 2.1.2 and § 2.1.3 movement logics with queueing on the same end prefers a job list where the majority of storage locations are situated closer to the order picking area. It is thus expected that these movement logic performs better compared to the rest than it did when analysed in § 5.1.

Shift	1	2	3	4	5	6	7	8	9	10
Front	12	29	21	19	20	21	20	20	33	14
Back	11	11	17	17	14	20	21	12	6	19

Table 5.8: Summary of the number of distributions that are present in the categories **front** and **back** for each storage rack. The summary excludes storage racks that have with no jobs and with an equal amount of **front** and **back** rack locations.

5.2.2 High lift count

High lifts are operated independent of each other when completing jobs within an aisle. The only time high lift movement is effected by one another is when high lifts change aisles. The effect of utilising one, two and four high lifts are investigated. Only up to four high lifts are investigated as PEP would realistically only ever use this many. Further, only an even number of high lifts are selected as this co-insides how highlifts start at the farthest aisles work towards each other (§ 1.8.2). By simulating the historical data of shift 2 the results are summarised in Table 5.9.

Number of high lifts	1	2	4
Average cumulative distance (rack locations)	15 814	15 891	15 941
Average cumulative dead heading distance (rack locations)	6 309	6 386	6 456
Average cumulative occupied distance (rack locations)	9 505	9 505	9 505
Average completion time (seconds)	62 296	30 578	15 862

Table 5.9: The effect adding more high lifts have on the average cumulative completion time, in seconds and distance, in rack locations.

Table 5.9 shows that by adding more high lifts to the storage rack system the average cumulative completion distance, in rack locations, does not increase drastically, 0.48% and 0.44% respectively. By adding more high lifts the dead-heading distances slightly increases, 1.22% and 1.09% respectively. The increase is because more high lifts need to move from their initial starting location, outside the storage rack system, to their starting queue and back again at the end of their shift. By increasing the number of high lifts the time at which all jobs are completed decreases. The number of high lifts, however, does not effect the total cumulative travel times and distances significantly.

5.2.3 Movement logic and algorithm comparison

Applying the same analysis used for the general job list the effects of the movement logics and algorithms are investigated for the historical date, received from PEP management. As an initial analysis the jobs performed in shift 2 are analysed and compared to the findings of the general job list analysis. Shift 2 is selected as it contains a sufficient number of jobs, in total and per aisle, to be able to do proper analysis. Table 5.10 summarises the number of each job type performed for each aisle. It is shown that shift 2 has a good variation between which job type is most dominant allowing for a better analysis of how the movement logics, algorithms and slotting configurations are effected. Aisles with less than a total of 10 jobs are ignored since their results does not give enough usable information. In cases where too few jobs, less than 10, are performed the different movement logics perform similar, resulting in similar completion times, in seconds.

Aisle Name	# Storage jobs	# Retrieval jobs	Job configuration
AC	23	2	More Storage
AD	14	14	Equal
AE	16	5	More Storage
AG	0	13	More Retrieval
AJ	17	6	More Storage
AK	12	13	More Retrieval
AL	7	8	More Retrieval
AN	12	9	More Storage
AU	8	7	More Storage
AV	9	27	More Retrieval
AX	6	8	More Retrieval
AY	11	11	Equal
AZ	22	9	More Storage
BA	8	6	More Storage
BB	21	4	More Storage
BC	18	3	More Storage
BD	12	5	More Storage
BE	20	2	More Storage
BF	12	13	More Retrieval
BG	15	2	More Storage
BO	0	12	More Retrieval

Table 5.10: Aisle break down of job types for shift 2

Each aisle is independently simulated for the different movement logics, sequencing algorithm and slotting configurations. The number of simulation repetitions for each aisle is calculated using equation (5.1), where the population consists of the completion times, in seconds.

As an first analysis, shift 2 is simulated by applying only the different movement logics. The average completing times, in seconds, are summarised in Table 5.11(a). For each aisle the underlines completion time, in seconds, represent the best movement logic. Investigating Table 5.11(a) either the SSP or CS movement logics performed the best, with less than 0.5% difference between the two movement logics. From the analysis performed on the general job lists this result is as expected. An interesting observation for aisle AG and BO, only one job type is performed, is that all five movement logics perform on average equally well, all results are within 1% of each other. This is because when no pairings can be formed the pairing movement logics follow the same logic as their unpaired counter parts. From Table 5.11(a) it is shown that irrespective of the job type distribution either the SSP or CS movement logic can be applied to obtain the best results in terms of completion time.

Table 5.11(b) summarises the average completion distance, in rack locations, to complete the

jobs for each aisle independently. The best performing movement logic is indicated for each aisle by underlining the shortest completion distance, in rack locations. For the majority of the aisles either the SSP or CS movement logics completed all jobs with the least distance (17 of 21 aisles). However, for aisles AC, AE and BD the OS movement logic performed the best - this is unexpected when compared to the completion times.

	SSP	OSP	OS	SS	CS		SSP	OSP	OS	SS	CS
AC	3 956	4 079	4 090	4 095	3 969	AC	938	959	916	969	937
AD	2 849	3 870	3 591	3 623	2 853	AD	719	1 068	936	1 071	722
AE	2 869	3 012	2 892	3 119	2 862	AE	808	607	476	865	806
AG	1 112	1 113	1 110	1 116	1 116	AG	319	316	316	319	320
AJ	2 896	3 489	3 529	3 334	2 890	AJ	676	1 041	1 054	805	677
AK	2 486	3 359	3 090	3 012	2 482	AK	517	918	814	767	519
AL	1575	2 054	1 939	1 879	1 582	AL	411	654	588	505	412
AN	2 154	2 917	2 728	2 581	2 144	AN	392	771	686	523	393
AU	1 509	2 072	1 982	1 898	1 507	AU	309	556	514	481	310
AV	3 499	4 033	3 795	3 955	3 510	AV	1 116	1 250	1 182	1 361	1 122
AX	1 470	1 883	1 809	1 777	1 477	AX	381	572	582	531	382
AY	2 282	3 014	2 793	2 836	2 289	AY	549	854	738	759	548
AZ	3 946	4 675	4 470	4 348	3 965	AZ	892	1 265	1 224	1 115	893
BA	1 496	2 035	1 990	1 818	1 494	BA	329	664	664	447	330
BB	3 472	3 887	3 788	3 695	3 488	BB	764	1 037	994	839	765
BC	3 053	3 213	3 186	3 216	3 046	BC	714	769	752	795	715
BD	2 201	2 469	2 353	2 532	2 204	BD	571	572	538	731	573
BE	3 135	3 389	3 382	3 298	3 122	BE	694	857	844	715	693
BF	2 603	3 372	3 120	3 246	2 590	BF	697	1 002	896	1 029	698
BG	2 219	2 403	2 357	2 333	2 209	BG	518	655	642	549	520
BO	1 082	1 087	1 077	1 074	1 078	BO	299	330	330	299	300

(a) Average completion times, in seconds

(b) Average completion distance, in rack locations

Table 5.11: Summary of average completion times, in seconds, and completion distances, in rack locations, when applying the five movement logics to the jobs performed in shift 2 for each aisle. The underlined values represent the movement logic that performed the best for the respective aisle.

Aisles AC, AE and BD are investigated together. As an initial observation for all three aisles the number of storage jobs exceed that of the number of retrieval jobs. For these aisles the storage jobs are stored in rack locations closest to the queue furthest from the picking area. The retrieval jobs are retrieved from rack locations close to the queue closest to the pick area. This means that when the OS and OSP movement logics are applied the storage and retrieval jobs are performed close to their respective queues. However, for the SSP, SS and CS movement logics the storage job locations are very far, more than 20 rack locations, away from the storage queue. This allows the OS and OSP movement logics to perform better than the other movement logics. Applying the OSP movement logic, however, forces the high lift pair an retrieval job after each storage jobs. With all the retrieval jobs being situated on the opposite end of the aisle the pairing causes a lot of extra dead heading. This dead heading results in the OSP always performing worse than the OS even with pairing. The results in Table 5.11(b) show that best results are obtained when the SSP and CS movement logics are applied except when the storage jobs need to be stored far from their storage queue and there are few opportunities for job pairing to be formed.

The next analysis is performed by again simulating the jobs performed in shift 2 applying the five movement logics together with a sequencing algorithm. The Hungarian sequencing algorithm is used as Table 5.4 shows that this algorithm achieves the best solution in the shortest

computational time. When simulating the CS movement logic the “Pairing from best” sequencing algorithm is applied. This algorithm also reaches the best solution only after a longer computational time.

Table 5.12(a) summarises the average completion times, in seconds, for the five different movement logics with a sequencing algorithm applied. The underlined values represent the movement logic that performed the best for the respective aisle. With a sequencing algorithm applied the SSP and CS movement logics again perform the best for all aisles. This is expected as the sequencing algorithm aims to reduce the completion time and will never accept a worse sequence than the initial sequence. This means that when a sequencing algorithm is applied the result will at least be as good as when no sequencing algorithm is applied. As with the general job list the decrease in completion time is low, on average 1.14% and in the best case 4.64% (Table D.7(a)).

Table 5.12(b) shows the average completion distance, in rack locations, for each aisle when a sequencer algorithm is applied. It is shown that the OS movement logic performs the best for aisles AC, AE and BD - the same as when no sequencer is applied. Even though the best sequence is executed too few job pairings can be formed in each aisle. The distance saved by forming proper pairings is not enough to reduce the total travel distance due to the storage jobs being situated so far from the storage queue.

	SSP	OSP	OS	SS	CS		SSP	OSP	OS	SS	CS
AC	<u>3 946</u>	4 070	3 992	4 067	3 961	AC	938	907	<u>898</u>	959	934
AD	<u>2 837</u>	3 854	3 526	3 623	2 840	AD	<u>629</u>	978	878	1 017	632
AE	<u>2 803</u>	2 951	2 890	3 091	2 811	AE	808	536	<u>438</u>	847	812
AG	<u>1 106</u>	1 107	1 109	1 107	1 111	AG	319	316	<u>316</u>	319	320
AJ	2 834	3 470	3 491	3 300	<u>2 828</u>	AJ	<u>526</u>	1 001	1 014	765	527
AK	<u>2 464</u>	3 333	3 072	2 973	2 476	AK	435	836	812	711	<u>434</u>
AL	1 537	2 007	1 922	1 853	<u>1 533</u>	AL	<u>305</u>	548	566	459	306
AN	2 146	2 885	2 720	2 565	<u>2 136</u>	AN	<u>372</u>	721	684	507	373
AU	1 505	2 065	1 975	1 872	<u>1 504</u>	AU	<u>296</u>	543	486	437	297
AV	3 405	3 956	3 704	3 930	<u>3 395</u>	AV	937	1 070	1086	1 323	<u>935</u>
AX	<u>1464</u>	1 866	1 809	1 757	1 468	AX	359	550	578	519	<u>358</u>
AY	2 258	2 989	2 783	2 801	<u>2 254</u>	AY	<u>469</u>	774	730	757	470
AZ	3 786	4 642	4 445	4 346	<u>3 781</u>	AZ	785	1 206	1 208	1 059	<u>782</u>
BA	1 469	2 001	1 980	1 802	<u>1 463</u>	BA	<u>301</u>	659	646	415	302
BB	3 466	3 814	3 786	3 684	<u>3 460</u>	BB	728	992	990	795	<u>727</u>
BC	<u>3 015</u>	3 205	3 181	3 199	3 023	BC	<u>704</u>	753	748	765	706
BD	2 179	2 413	2 337	2 471	<u>2 174</u>	BD	549	521	<u>500</u>	691	550
BE	3 117	3 387	3 360	3 282	<u>3 112</u>	BE	<u>676</u>	845	832	711	677
BF	2 593	3 325	3 098	3 226	<u>2 587</u>	BF	<u>565</u>	870	884	977	567
BG	2 209	2 389	2 338	2 332	<u>2 199</u>	BG	474	655	628	545	<u>472</u>
BO	1 069	1 078	1 073	1 068	<u>1 071</u>	BO	<u>299</u>	330	330	299	300

(a) Average completion times, in seconds

(b) Average completion distance, in rack locations

Table 5.12: Summary of cumulative completion times, in seconds, when applying the five movement logics in unison with sequencing algorithms to the jobs performed in shift 2 for each aisle. The underlined values represent the movement logic that performed the best for the respective aisle.

Unlike with the completion times the average completion distance decrease substantially when a sequencing algorithm is applied. On average the total distance for each aisle is decreased by 9.38% with the biggest decrease being in aisle AL, 25.79% (Table D.7(b)). It would be expected that the average completion time and distance would improve similarly - but this is not the case. It is investigated why applying a sequencing algorithm effect the completion time less than the

Due to the high volume of RC1 pallets that flow into the storage racks this RC value gets a dedicated row. Having entire rows dedicated to a single RC value leads to pallets with certain RC values not having adequate space for placement when the storage rack system is simulated. This occurrence is especially apparent in cases where the storage racks are filled close to capacity — it is 80% and higher. This lack of space for a specific RC value leads to pallets not being stored or stored in a different storage rack. Both these result in longer completion times. Shared storage

is thus essential to prevent pallets with a certain RC value from not being stored. Although shared storage is used pallets with lower RC values still receive the best locations. Pallets with higher RC values are only stored in the same row as pallets with lower RC values when there is a shortage of space.

Shift 2 is simulated gain by applying the different movement logics together with the sequencer algorithm and slotting algorithm. The average completing times, in seconds, and distance, in rack locations, are summarised in Table 5.13. For each aisle, the underlined values represent the best movement logic.

	SSP	OSP	OS	SS	CS		SSP	OSP	OS	SS	CS
AC	3656	3809	3778	3787	3652	AC	410	500	464	423	412
AD	2725	3757	3375	3326	2733	AD	477	976	570	597	476
AE	2321	2723	2543	2537	2329	AE	187	433	246	219	188
AG	1107	1109	1103	1106	1102	AG	319	316	316	319	320
AJ	2651	3042	2924	2933	2645	AJ	376	534	424	445	377
AK	2327	3194	2738	2760	2317	AK	383	814	428	467	384
AL	1495	1945	1773	1777	1497	AL	297	480	312	343	296
AN	2028	2743	2336	2382	2031	AN	219	653	264	287	220
AU	1370	1941	1823	1822	1375	AU	229	524	284	301	228
AV	3348	3909	3651	3685	3333	AV	917	1042	952	971	919
AX	1421	1803	1671	1674	1414	AX	357	454	374	401	358
AY	2103	2916	2545	2531	2111	AY	383	762	438	463	382
AZ	3823	4499	4270	4320	3816	AZ	532	824	638	651	533
BA	1470	1926	1767	1777	1472	BA	300	448	366	345	301
BB	3417	3663	3594	3587	3409	BB	410	567	472	485	411
BC	2877	3128	3082	3103	2891	BC	361	444	374	401	360
BD	2051	2415	2260	2293	2046	BD	272	440	306	325	273
BE	3044	3158	3125	3120	3037	BE	287	379	332	307	286
BF	2501	3293	3062	3050	2510	BF	535	850	632	655	533
BG	2108	2211	2151	2192	2076	BG	283	327	312	307	282
BO	994	1018	1002	1012	998	BO	299	330	330	299	300

(a) Average completion times, in seconds

(b) Average completion distance, in rack locations

Table 5.13: Summary of cumulative completion times, in seconds, when applying the five movement logics in unison with sequencing and slotting algorithms to the jobs performed in shift 2 for each aisle. The underlined values represent the movement logic that performed the best for the respective aisle.

From the results it is clear that for all aisles using either the SSP or CS movement logics the best results are obtained. Further more by applying both a sequencing and slotting algorithm the average completion time, in seconds is decreased by an average of 8.91%, in the best case an decrease of 18.66% is observed (Table D.8(a)). Investigating the completion distances, in rack locations, reveal significant decreases with an average of 44.88%, with a decrease of 76.85% in the best case (Table D.8(b)).

Unlike the previous two cases, the SSP and CS movement logics perform the best for all aisles. This is because the slotting algorithms changes the storage locations to the location best suited for each movement logic. This negates the problem encountered in the previous cases where the the storage jobs are far from the storage queue for some movement logic. Combining the shorter distance to storage locations and pairing of jobs the SSP and CS movement logics perform better than the other movement logics.

The analysis on shift 2 shows that by applying the SSP or CS movement logic to the historical data results in the fastest completion time and shortest completion distance. Completion time

and distance can further be decreased by applying a sequencing algorithm - reducing completion time by 0.92% and completion distance by 9.97%. By applying a slotting algorithm that utilises a Row slotting configuration the completion time is decreased by 3.87% and completion distance by 28.91%.

As final attempt to find the movement logic capable of obtaining the best result the historical data from 14-Nov-2015 to 19-Nov-2015 is simulated. The simulated results are compared to the historical completion time and distance to determine how well applying different movement logics and algorithms can improve on PEP's current system.

5.2.4 Historical completion time and distance

Using the knowledge obtained from simulating general job lists and the jobs of shift 2, all 10 shifts are now simulated together. By simulating multiple shifts gives insight on how different movement logics and algorithms will affect PEP's storage rack operations in the long run. For this analysis the effect of having a number of simulations repetitions are calculated using equation (5.1). Figure 5.14 illustrates the average completion distance, in rack locations, obtained by simulating the jobs exactly as it is in the historical data (PEP_S) and simulating the five proposed movement logics. As expected the total occupied travel distances for the different logics that have the same queueing configuration are equal.

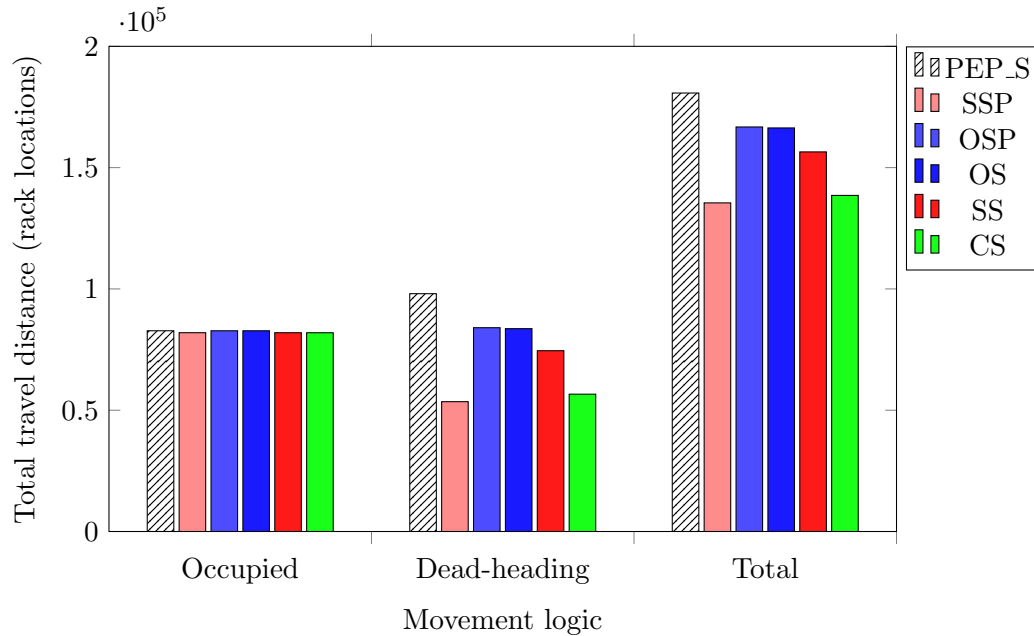


Figure 5.14: Total cumulative distance for high lifts to complete the same set of jobs using different movement logics.

It is expected that the total dead-heading distance for the PEP_S and OS simulations to have similar results. The OS movement logics is formulated using the description given by PEP [23] and should thus work similarly. Comparing the dead-heading distance of the two results shows a difference of 15.88% between them. The historical job list is investigated to determine why the difference is so large. It is found that the order in which jobs are performed has the consequence of increasing the dead-heading distance.

Table 5.14 shows an example of the order in which jobs are sometimes performed in the historical data. In the example High lift 2 starts with storing pallets in storage rack AC. The high lift

perform six storage jobs in this storage rack before moving to the next. Four pallets are now stored in storage rack AE before the high lift moves back to storage rack AC where it performs five storage jobs. The high lift then stores a pallet in storage rack AD before storing the last pallet destined for storage rack AC. This means that a total of four unnecessary storage rack/aisle change are completed, each taking 20-140 seconds, before all jobs are stored. The increase in total number of storage rack/aisle changes contributes to the large difference in dead-heading distance and time between simulating the job list exactly and using the OS movement logic.

TYPE	LOCATION	ID	START_DATE	START_TIME	HIGH LIFT NUM
IN	AC05009	3521545	14/11/2015	09:54:20	2
IN	AC05015	3521554	14/11/2015	09:56:01	2
IN	AC05008	3521553	14/11/2015	09:57:34	2
IN	AC04027	3521546	14/11/2015	09:59:32	2
IN	AC04024	3521547	14/11/2015	10:00:59	2
IN	AC04022	3521549	14/11/2015	10:02:30	2
IN	AE05024	3520628	14/11/2015	10:09:13	2
IN	AE02025	3523344	14/11/2015	10:16:27	2
IN	AE02019	3523343	14/11/2015	10:20:39	2
IN	AE02026	3523342	14/11/2015	10:24:02	2
IN	AC05016	3521555	14/11/2015	10:26:24	2
IN	AC05004	3521083	14/11/2015	10:28:51	2
IN	AC05017	3521084	14/11/2015	10:30:41	2
IN	AC05021	3521560	14/11/2015	10:32:37	2
IN	AC05020	3521558	14/11/2015	10:33:56	2
IN	AD03027	3522591	14/11/2015	10:35:27	2
IN	AC05005	3521551	14/11/2015	10:36:31	2

Table 5.14: Example of how the historical data does not follow the flow described by PEP.

Table 5.15 summarises the percentage each movement logic decreases the average completion distance, in rack locations, compared to the historical job list. The table shows that no matter which movement logic is applied the occupied distance does not change significantly, it is less than 1%. However, the biggest reduction in completion distance is observed to happen during dead-heading. All five movement logics succeed in reducing the dead-heading distance. The SSP movement logic achieved to greatest decrease in dead-heading with 58.75%. This finally equates to a total decrease in completion distance of 45.32 km.

	SSP	OSP	OS	SS	CS
Occupied	0.968	0	0	0.968	0.984
Dead-heading	58.75	15.41	15.88	27.24	53.57
Total	28.64	8.05	8.29	14.39	26.44

Table 5.15: The percentage each movement logic improves on the distance of simulating the job list exactly.

By investigating the travel times, further evidence is gathered to find the movement logic that improves on the current system the most. Figure 5.15 illustrates the cumulative travel time it took each movement logic to complete the entire job list. Figure 5.15 closely resembles the cumulative distance in Figure 5.14. This is expected as the travel time is a function of the travel distance. It thus stand to reason that the explanations why results may not be as expected is the same as with the cumulative travel distance.

In conjunction to visually interpreting the results, the improvement percentage each movement

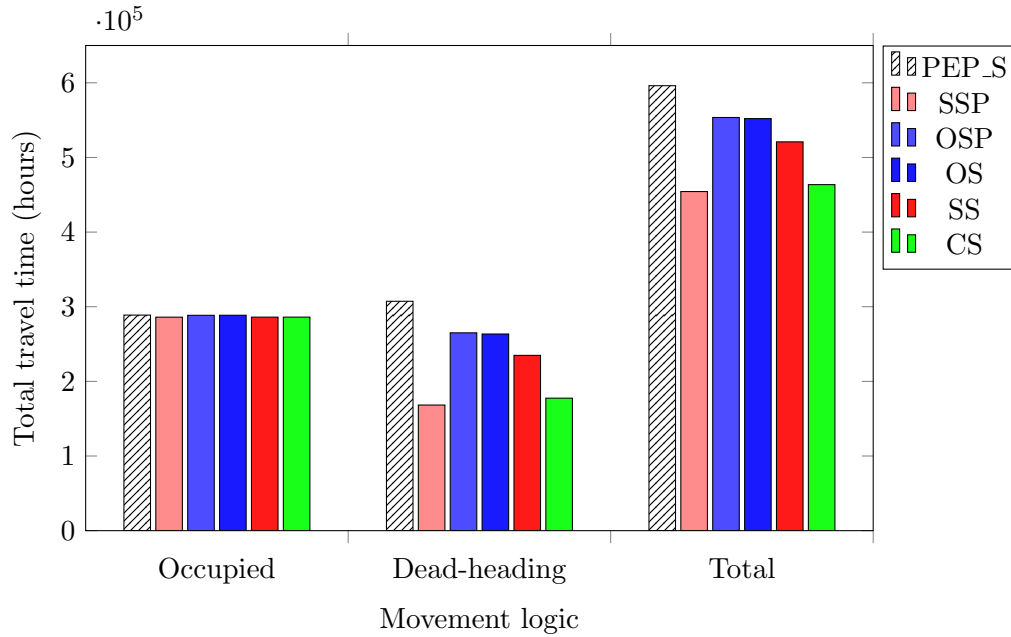


Figure 5.15: Total cumulative time for high lifts to complete the same set of jobs using different movement logics.

logic has on the historical completion time is investigated. Table 5.16 summarises the improvement percentage each proposed movement logics has, in cumulative time, when compared to the historical completion time. As expected these values closely resemble those found in Table 5.15. It is clear that the SSP movement logic improves the current system the most by decreasing the completion time with 23.78%, 39:23:13 hours.

	SSP	OSP	OS	SS	CS
Occupied	0.968	0.094	0.073	0.953	0.952
Dead-heading	58.46	14.78	15.37	26.74	53.54
Total	27.00	7.40	7.68	13.47	25.02

Table 5.16: The percentage each movement logic improves on the time of simulating the job list exactly.

Since the SSP movement logic reduced the completion time, in seconds, the most its productivity is compared to the historical data. Inspecting the average number of jobs completed per hour (Figure 5.16) it is shown that in 74% of the cases the SSP movement logic completes more jobs per hour than the current system. It takes the SSP movement logic 50:33:36 hours to complete the entire job list while the current system took 84:18:25 hours to complete the same job list - ignoring shift changes and breaks. Considering the average number of jobs completed per hour the difference in efficiency between the SSP movement logic and the current system is much clearer. It is important to note that these completion times do not represent the cumulative time to complete all jobs but rather the time span over which they were completed.

Due to the improvement being so small when only adding the sequencer algorithm (§ 5.2.3) it is only analysed together to the slotting algorithm. To determine the effect of applying the **Row pallet slotting** placement algorithm has on the current system it is first simulated with an almost empty storage rack system - only pallets that are retrieved during the simulation is initially stored. This results in finding the best case scenario for the total completion distance, in rack locations and time, in seconds, needed to complete the historical job list.

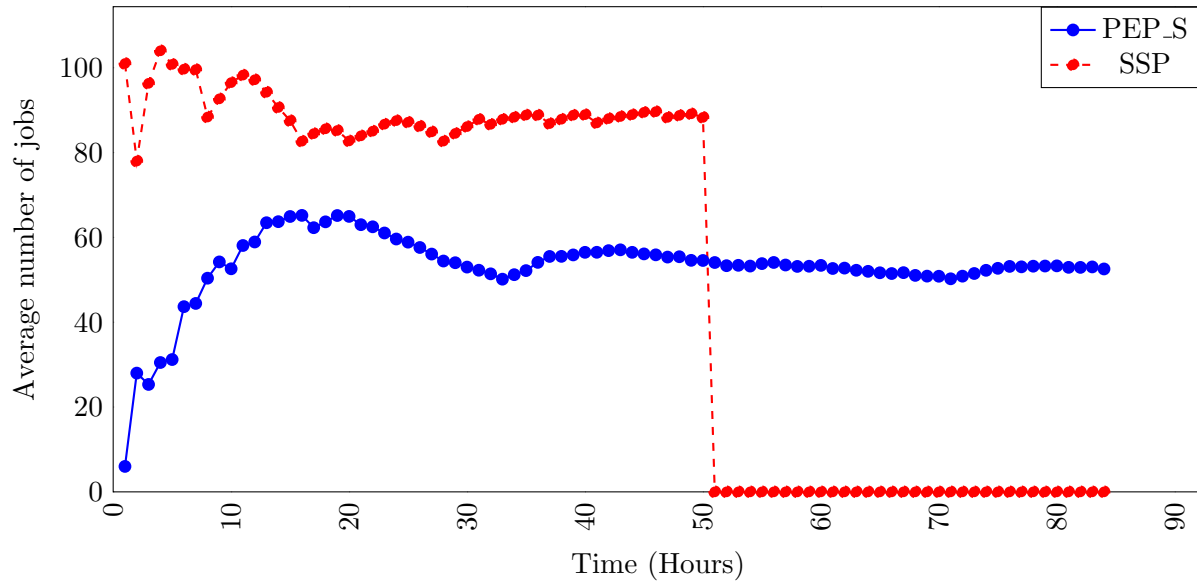


Figure 5.16: Graphical comparison of the average number of jobs performed per hour between the historical data and using the SSP movement logic.

Both the slotting and sequencing algorithms are applied to each of the five movement logics and simulated. Figure 5.17 illustrates a comparison between the five movement logics and the historical travel distances. It is clear that when using the two algorithms in conjunction with the movement logics the reduction in travel distance is exceptional, 46.52%, 31.69%, 38.62%, 42.18%, 44.49%, respectively. The OSP movement logic, which perform the worst of the five, improves on the PEP_S by 31.69%.

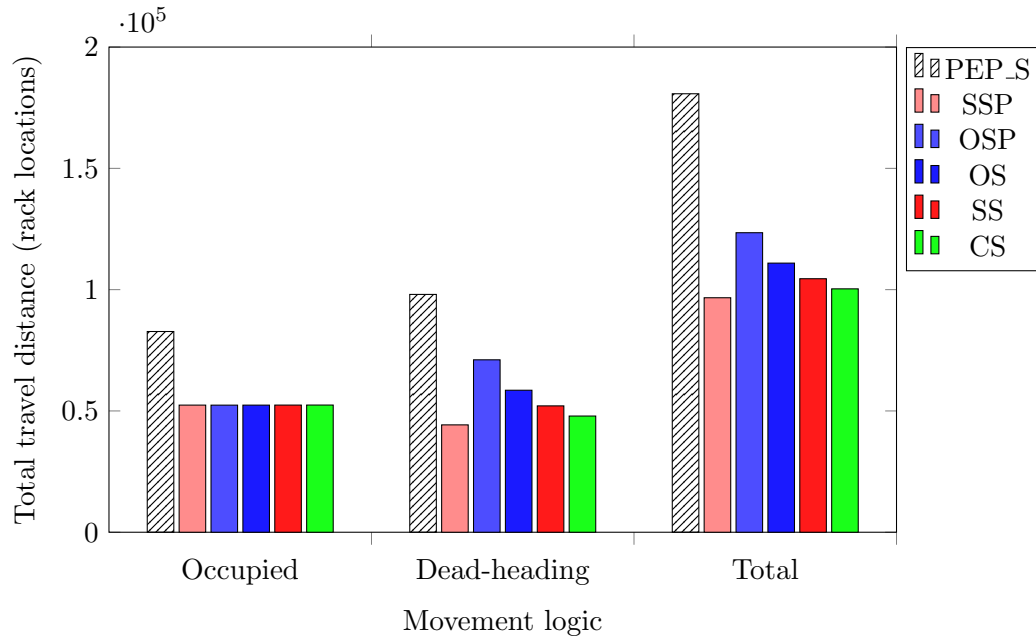


Figure 5.17: Total cumulative distance for high lifts to complete the same set of jobs using the row pallet slotting placement algorithm and five different movement logics.

The SSP movement logic reduces the total travel distance the most by improving on PEP_S with 46.52%. This means that in a best case scenario if a SSP high lift movement logic is used

in conjunction with a sequencing and slotting algorithm, it is possible to reduce the total travel distance with 84.1 km.

These results are, however, obtained assuming the storage rack system is almost empty. The bulk of the pallets are stored in the storage rack system S1 and is almost always at 90% capacity, according to Ted Moodley [23]. In order to replicate the real world system as accurately as possible the storage rack system is filled with static pallets when the placement algorithm is used. Static pallets are added to the storage rack to ensure after all pallets are stored and retrieved the storage rack system is filled to 90% its capacity. Applying the sequencing and slotting algorithms again Figure 5.18 is constructed and visually inspected to analyse the effect of using a storage rack system filled to 90% capacity has on the combinations of algorithms and logics.

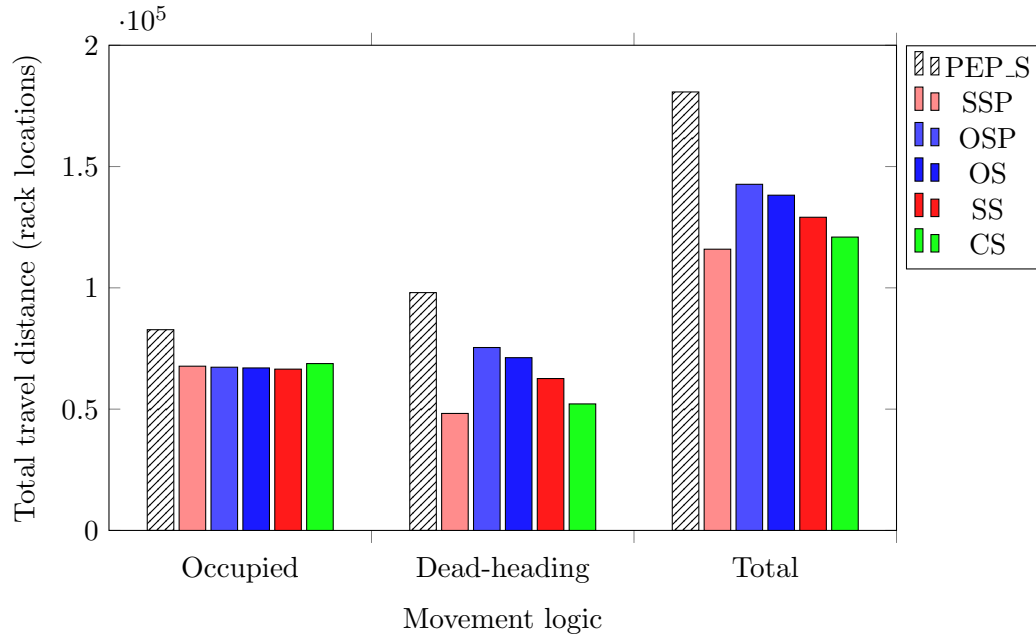


Figure 5.18: Total cumulative distance for high lifts to complete the same set of jobs using the row pallet slotting placement algorithm and five different movement logics.

Table 5.17 shows that when the rack is filled to 90% of its capacity the results are worse than when the storage rack is almost empty. The total movement distance, in rack locations, increases by 19.93%, 15.56%, 24.57%, 23.53%, 20.53% respectively for each of the five movement logics compared to when an empty storage rack system was used.

	PEP	SSP	OSP	OS	SS	CS
Empty	180749	96663	123480	110638	104505	100329
90% full	180749	115929	142684	138191	129098	120928
Increase %	0	19.93	15.55	24.9	23.53	20.53

Table 5.17: Summary of how the completion distance, in rack locations, increases when the storage racks is filled to 90% capacity compared to when it is empty.

It is expected that the results will be worse as more pallets are stored in the storage rack system. The algorithm is formulated to find the best storage location from the storage queues. This means that as the storage rack system becomes fuller more of the storage pallets need to be placed further from the storage queue. This in turn increases the travel distance, seen when comparing Figure 5.17 and Figure 5.18.

The same analysis is performed to find the average completion time, in seconds, when the storage rack system is empty en 90% full. The results are summarised in Table 5.18. Again there is a increase from when the jobs are simulated using an almost empty storage rack system compared to when it is 90% full. This increase is, however, not as big as with the completion distance, biggest increase is 4.83%.

	PEP	SSP	OSP	OS	SS	CS
Empty	596118	419272	512776	511213	480009	436746
90% full	596118	440695	536951	541006	510481	454303
Increase %	0	3.49	3.4	3.45	4.76	4.83

Table 5.18: Summary of how the completion time, in seconds, increases when the storage racks is filled to 90% capacity compared to when it is empty.

It is thus advised to change PEP's current storage and retrieval logic to a paired job with queueing on the same side movement logic. To further improve on this high lift movement logic it is advised that a row pallet slotting algorithm is utilised in conjunction with the movement logic. If this combination is used in the DC it is possible to reduce the total travel distance with 35%, 64.82 km, and the time needed to complete the job list with 43:03:00 hours.

CHAPTER 6

Conclusion

The primary objective of the study is to accurately simulate all combinations of high lift movement logics, job sequencing algorithms and pallet slotting algorithms within the storage rack system for PEP's Durban distribution centre (DC). A simulation model is coded with a high level of detail that is capable of achieving this objective.

However, before the simulation model can be utilised to improve on the system currently employed by PEP all aspects of current system must first clearly be understood. Consultation with PEP's DC line manager, Ted Moodley [23], revealed that the underlining problem PEP DC is facing is a bottleneck being formed in the storage rack system. This bottle neck is formed because the DC is capable of process an higher inflow of pallets into the DC, that need to be stored, than the rate at which the DC is currently capable of retrieving pallets from the storage racks. The reason why the outflow of pallets is low relative to the inflow rate is due to the long travel distances associated with completing jobs. Further the logics and rules of movement for the current system is explained in detail to understand exactly what the system is capable of.

Using this gained knowledge regarding the functionality of the high lifts used in the storage rack system five alternative high lift movement logics are developed. These newly formulated movement logics aim to reduce the time and distance needed to complete the process of retrieving and storing pallets.

The movement logics are further improved by introducing a sequencer algorithm. The purpose of this algorithm is to determine the optimal sequence in which storage and retrieval jobs should be performed. Depending on which movement logic is used the method of sequencing jobs and the effectiveness of the algorithm changes. Five sequencing algorithms are developed of which four are meta-heuristic approaches and the remaining one is an allocation algorithm.

In an final attempt to further increase the flow rate of pallets in the storage rack system the storage location of pallets are investigated. Considering the length of a pallet's replenishment cycle (RC) optimal regions are assigned to the storage racks for each RC value. Following the advice of Bartholdi [2] the RC regions are configured in such a way to allow the fastest moving pallets to be stored in the most convenient locations. Five RC region configuration are created based on distance or from logical analysis.

Simulating a real world system exactly is nearly impossible. Using all the information gathered regarding the processes in the DC logical assumptions are established to simplify the system and make it possible to simulate. Using the assumptions and gathered information regarding the DC a simulation model is implemented using Python 3.4. This simulation model, comprised of agent based models, is used to replicate the behaviour of high lifts in the DC as well as simulating the

effect of different high lift movement logics, sequencing algorithms and slotting algorithms have on the DC.

The accuracy of the simulation model is verified and validated according to the methods given by Robinson, Naylor and Finger. All aspects of the simulation is tested thoroughly by rerunning the simulation multiple times using generated job list. After each simulation it is confirmed that the simulation model interprets the jobs in the correct format, calculates the travel distance and time accurately, reads and performs the jobs in the correct sequence, stores and retrieves pallets from the correct location and applies the logics and algorithms effectively.

Before any comparable result can be obtained from the simulation a job list is created using the historical data received from PEP. The data is gathered at PEP through their WMS and stored in a usable format. In order to generate an job list the data must first be cleaned. All unusable data points are removed from the data set and is verified and validated to ensure that it accurately portrays the real world system. Analysis is conducted on timestamps of the historical data to determine appropriate travel times into and out of the storage racks. The distributions of each storage location is determined and compared to known distributions. It is found that these distribution do not fit any of the known distributions. A function is formulated to generate accurate travel times using each storage location's unique distribution. The historical data does not give clear indication as to which high lift performed each job. A method is designed using the given employee ID's to identify the high lift used by each employee. Lastly the historical data does not contain RC values for each pallet. An algorithm is thus created that is capable of assigning RC values to each job based on how long pallets are on average stored in the storage racks.

From the data set a job list is compiled spanning a time interval of five consecutive days. The days are selected to contain a relatively large number of data points while requiring as little as possible clean up. Multiple simulation are tested using the generated job lists to validate the data and further validate the simulation model.

Using the simulation model it is possible to repeatedly and effectively simulate different combinations of high lift movement logics, sequencer algorithms and slotting algorithm. After each simulation the total travel time and distance are stored and analysed. From the totals it is possible to determine the total dead heading times and distances. Analysis reveals how each combination effects these values and allows for the most efficient combination to be identified.

6.1 Recommendations

With the current high lift logic utilised at PEP's DC an average cumulative total high lift travel time of 165:35:24 is needed to complete the selected historical data. The total cumulative distance travelled by the high lifts for PEP's current system is $\pm 180\,749$ km. Changing from the current movement logic to the paired job with same end queue movement logic (SSP) reduces the travel times and distance. Utilising this combinations reduces the total travel time with 39:23:13 and the total travel distance is decreased with 45.27 km to $\pm 135\,463$ km.

The system is further improved by adding the row pallet slotting algorithm and sequencer algorithm together with the SSP movement logic. By utilising this combination the travel distance is further decreased with 19.53 km meaning in total, using this combination reduced travel distance by 64.80 km. The total completion time is decreased by 43:03:00 when applying this combination. It is thus recommended that this combination be implemented in PEP's Durban DC.

If this combination is selected it comes with some disadvantages. With this movement logic the

storage and retrieval queues are on the same end of the storage racks and might lead to confusion as to which pallets are destined for the storage racks or order picking area. The queues can also become very long since both are on the same end. As stated in §1.8.2 the length of the storage and retrieval queues should be held at minimum.

According to the high lift travel time analysis it is recommended that pallets rather be stored closer to the ground level than higher up in the storage racks.

6.2 Future work

The most crucial part of this study is finding and using an accurately job list. The job list generated in this study can further be improved by observing the processes in the DC. Observing the movement of all movement equipment within the storage rack system an even better understanding of the processes performed can be obtained. This deeper understanding can allow reason why jobs are not performed exactly as expected in the historical job list could be determined. Further an explanation for the unusual trend for the travel times of storage jobs performed on the ground level could be determined. It could even be beneficial to record travel times physically in the DC by timing high lifts movements to different rack locations and aisles. These travel times can then be compared to the travel times deducted from the historical data. This can form part of a deeper validation of the historical data.

A bigger job list could be compiled from the historical data. With the current job list very few pallets are simulated to be stored and retrieved. Having a job list with more jobs would allow for more jobs to be stored and retrieved allowing for better analysis on the effect of each logic and algorithm.

As an extension to the current scope the assumption that queues can become infinitely long can be removed. This would allow for the effect of each logic and algorithm on the queue lengths to be analysed. This analysis would assist in determining the necessary storage space needed at the end of each storage rack. Having this knowledge could shed light on how the layout of the storage rack system needs to be changed to support the change in movement logic and placement configuration.

As seen in the analysis of shift 2 different movement logics perform better with certain job list configuration. It could thus be worth investing in a system that is capable of dynamically analysing each job list and assigning a movement logic based on the job list configuration. This addition opens up multiple new problems to analyse - like how this would affect the queue locations or logistics of how high lifts move. This will, however, only be applicable for cases where only different movement logics and a sequencer algorithm are applied. When the slotting algorithm is applied the job list configuration is altered to best suite each movement logic.

In this study only one type of high lift is tested. Testing multiple makes and models of high lifts would mean different travel times are used. Using different travel times could reveal more insight into which movement logic and algorithm combinations work best for different horizontal and vertical movement speeds.

List of References

- [1] AYERS JB, 2006, *Handbook of supply chain management*, Series on resource management, 2nd Edition, Auerbach publications, Taylor and Francis group, New York, USA.
- [2] BARTOLDI J & HACKMAN S, 2011, *Warehouse and distribution science*, Georgia Institute of Technology, Available from [Availableatwww.warehouse-science.com](http://www.warehouse-science.com).
- [3] BREMENPORTS, *Expert transhipment of sensitive goods*, Available from <http://www.bremenports.de/en/location/the-strengths/refrigerated-goods>, [Online], [Cited November 28, 2016].
- [4] BURGHOUT W, 2004, *A note on the number of replications runs in stochastic traffic simulation models*, Centre for Traffic Research, Stockholm, Sweden.
- [5] BURKE E & KENDALL G, 2006, *Introductory Tutorials in Optimization and Decision Support Techniques*, Springer Science and Business Media, New York, USA.
- [6] CARSON J, 2002, *Verification validation: model verification and validation*, Proceedings of the Winter simulation conference, IEEE Computer Society, 2002, San Diego, USA.
- [7] CHIN WW, 1998, *The partial least squares approach to structural equation modeling*, Modern methods for business research, **295(2)**, pp. 295–336.
- [8] DRAPER NR & SMITH H, 2014, *Applied regression analysis*, John Wiley & Sons, New Jersey, USA.
- [9] EDMUND K BURKE GK, 2013, *Future Paths for Integer Programming and Links to Artificial Intelligence*, 2nd Edition, Springer, New York, USA.
- [10] EHLERS A, 2008, *Renier van Rooyen and PEP stores limited: The genesis of a South African entrepreneur and retail empire*, South African Historical Journal, **60(3)**, pp. 422–451.
- [11] FRAZELLE E, 2001, *World-class warehousing and material handling*, 1st Edition, McGraw-Hill, New York, USA.
- [12] GUNES, *Verification and validation of simulation models*, Available from http://www.mi.fu-berlin.de/inf/groups/ag-tech/teaching/2012_SS/L_19540_Modeling_and_Performance_Analysis_with_Simulation/10.pdf, [Online], [Cited June 22, 2017].
- [13] JUNGHEINRICH, *Ekk 410*, Available from http://www.jungheinrich.com/uploads/jh_importer/assets_product_5803_en-ZKW___pdf___link/EKX_410_data_sheet.pdf, [Online], [Cited November 11, 2016].

- [14] KELTON D, SADOWSKI R & SWETS N, 2002, *Simulation with Arena*, 5th Edition, McGraw-Hill, New York, USA.
- [15] KENGPOLO A & O'BRIEN C, 2001, *The development of a decision support tool for the selection of advanced technology to achieve rapid product development*, International Journal of Production Economics, **69**(2), pp. 177–191.
- [16] KUHN HW, 1955, *The hungarian method for the assignment problem*, Naval Research Logistics Quarterly, **2**, pp. 83–97.
- [17] LAHMAR M, 2007, *Facility Logistics: Approaches and solutions to next generation challenges*, Auerbach Publications, Baco Raton, USA.
- [18] LAW AM, 2014, *How to select simulation input probability distributions*, Proceedings of the Proceedings of the Winter Simulation Conference, pp. 1394–1407.
- [19] LAW AM, KELTON WD & KELTON WD, 1991, *Simulation modeling and analysis*, volume 2, McGraw-Hill, New York, USA.
- [20] MARIA A, 1997, *Introduction to modeling and simulation*, Proceedings of the 29th conference on Winter simulation, 1997, Atlanta, USA, pp. 7–13.
- [21] MCINTOSH C & IDM, 2013, *Cambridge Advanced Learner's Dictionary*, 4th Edition, Cambridge University Press, Cape Town, RSA.
- [22] MINITAB, *Minitab 17*, Available from <http://www.minitab.com/en-us/products/minitab/>, [Online], [Cited August 25, 2016].
- [23] MOODLEY T, 2014, *Warehouse manager*, contactable at: dbndbn1@pepstores.com.
- [24] MWPL INTERNATIONAL, *The art and science of warehouse slotting optimization*, Available from http://http://www.mwpl.com/html/warehouse_slotting_optimization.html, [Online], [Cited November 20, 2017].
- [25] NAYLOR T, BALINTFY J, BURDICK D & CHU K, 1967, *Computer Simulation Techniques*, volume 2, John Wiley & Sons, New Jersey, USA.
- [26] NAYLOR TH, FINGER JM, MCKENNEY JL, SCHRANK WE & HOLT CC, 1967, *Verification of computer simulation models*, Management Science, **14**(2), pp. pp. B92–B106, Available from <http://www.jstor.org/stable/2628207>.
- [27] OMIUSAJPIC, *Supply chain*, Available from <http://omiusajpic.org/issues/investing/supply-chain/>, [Online], [Cited November 25, 2016].
- [28] ÖREN T, 2001, *Impact of data on simulation: from early practices to federated and agent-directed simulations*, Tubitak-Marmara Research Center, Information Technologies Research Institute, Turkey.
- [29] PATIL MV & YOGI AN, 2011, *Importance of data collection and validation for systematic software development process*, Journal of Computer Science & Information Technology, **3**(2), pp. 260–277.
- [30] PEP, *Pep facts*, Available from www.pepstores.com/find-out-more/pep-facts/, [Online], [Cited November 27, 2016].
- [31] PEPKOR, *About pepkor*, Available from <http://www.pepkor.co.za/>, [Online], [Cited November 25, 2016].

- [32] PIASECKI D, *Warehouse management systems (wms)*, Available from http://www.inventoryops.com/warehouse_management_systems.html, [Online], [Cited February 12, 2017].
- [33] POTTER K, HAGEN H, KERREN A & DANNENMANN P, 2006, *Methods for presenting statistical information: The box plot*, Visualization of Large and Unstructured Data Sets, s, 4, pp. 97–106.
- [34] PRODUSERVICE, *Ekk 515*, Available from http://www.produservice.com.ar/images/trilaterales/EKX_515/EKX_515_01.jpg, [Online], [Cited November 27, 2016].
- [35] ROBINSON S, 1997, *Simulation model verification and validation: increasing the users' confidence*, Proceedings of the 29th conference on Winter simulation, IEEE Computer Society.
- [36] SARGENT RG, 2008, *Verification and validation of simulation models*, Proceedings of the 37th conference on Winter simulation, winter simulation conference, 2007, Washington, USA.
- [37] SCOTT DW, 2015, *Multivariate density estimation: theory, practice, and visualization*, John Wiley & Sons, New Jersey, USA.
- [38] SERDYN H, 2014, contactable at: henniese@pepkorit.com.
- [39] SPARKS JC, 2008, *The Pythagorean Theorem: Crown Jewel of Mathematics*, AuthorHouse, Indiana, USA.
- [40] TOMPKINS JA & SMITH JD, 1998, *The warehouse management handbook*, Tompkins press, Raleigh, USA.
- [41] TOYOTAEQUIPMENT, *Electric pallet jack*, <http://www.toyotaequipment.com/wp-content/uploads/2014/12/Toyota-electric-pallet-truck-rider-platform.jpg>, [Online], [Cited November 27, 2016].
- [42] UDOKA SJ, 1991, *Automated data capture techniques: A prerequisite for effective integrated manufacturing systems*, Computers and Industrial Engineering, **21(1)**, pp. 217 – 221, Available from <http://www.sciencedirect.com/science/article/pii/036083529190091J>.
- [43] WIKIMEDIA, *Modern retail warehouse*, http://upload.wikimedia.org/wikipedia/commons/a/a2/Modern_warehouse_with_pallet_rack_storage_system.jpg, [Online], [Cited November 27, 2016].
- [44] WINSTON WL, 2004, *Operations Research: Applications and Algorithms*, Cengage Learning, Kentucky, USA.
- [45] WISEGEEK, *What is a warehouse*, Available from <http://www.wisegeek.com/what-is-a-warehouse.htm>, [Online], [Cited November 20, 2017].
- [46] WONDERFULENGINEERING, *Inside an amazon warehouse*, Available from <http://wonderfulengineering.com/inside-an-amazon-warehouse/>, [Online], [Cited November 30, 2016].

APPENDIX A

Algorithms

A.1 Movement logics examples

To further explain each movement logic's equation an simple example is constructed. The storage rack system shown in Figure A.1 is used. This storage rack system is constructed with six rack column (x) and three rack levels (y). For the purpose of the example each rack location and pallet is a 1×1 m entity.

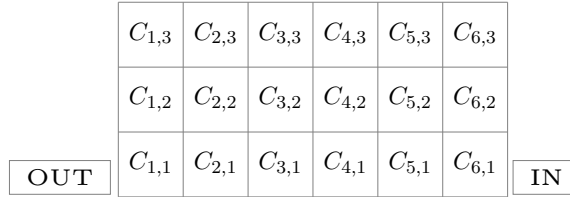


Figure A.1: A schematic representation of the storage rack system used in the example. The *IN* block refers to the storage queue and *OUT* block refers to the retrieval queue. The rack location of a pallet is denoted by $C_{x,y}$ where x is the rack column and y the rack level.

A.1.1 Single pallet movement types

A job list is generated to be used for all the example, summarised in Table A.1. To prevent adding complexity to the examples the job list is generated with an equal number of pallets that are stored and retrieved.

i	Movement type	x	y
1	Storage	6	3
2	Storage	6	2
3	Storage	5	3
4	Retrieval	1	3
5	Retrieval	2	1
6	Retrieval	2	2

Table A.1: Generated job list for the movement logics that moves one pallet per job.

In the generated example there three storage pallets ($I = 3$) and three retrieval pallets ($O = 3$). This makes the total number of jobs in the job list to be six ($K = 6$). Using this example the total movement distance a high lift travels is calculated for the two of the movement logics.

A.1.2 Single pallet with opposite side queueing

The first step is calculating the total distance travelled while a high lift is occupied, carrying a pallet. Substituting the relevant information equations (A.1) and (A.2) are obtained.

$$D_P = \sum_{i=1}^3 [N - x_i + y_i] + \sum_{i=4}^3 [x_i + y_i - 1] \quad (\text{A.1})$$

$$D_H = \sum_{i=1}^2 [N - x_i + y_i - 1] + \sum_{i=5}^2 [x_i + y_i - 1] + |x_3 - x_4| + |y_3 - y_4| \quad (\text{A.2})$$

Table A.2 summarises the distance a high lift moved to complete each job in the generated job list example. It is important to note that the calculation to determine the distance dead headed (D_H) when storing pallet 3 and retrieving pallet 4 are simultaneous.

i	Movement type	x	y	$D_P(\text{A.1})$	$D_H(\text{A.2})$
1	Storage	6	3	$6 - 6 + 3 = 3$	$6 - 6 + 3 = 3$
2	Storage	6	2	$6 - 6 + 2 = 2$	$6 - 6 + 2 = 2$
3	Storage	5	3	$6 - 5 + 3 = 4$	$ 5-1 + 3-3 = 4$
4	Retrieval	1	3	$1 + 3 - 1 = 3$	
5	Retrieval	2	1	$2 + 1 - 1 = 2$	$2 + 1 - 1 = 2$
6	Retrieval	2	2	$2 + 2 - 1 = 3$	$2 + 2 - 1 = 3$

Table A.2: Summary of the results obtained when substituting the generate job list example into the distance equation.

From the calculated distances summarised in Table A.2 it is possible to find the total occupied and dead heading distances the high lift travelled. Summing over the distances the high lift travelled a total of $D_P = 17$ m with a pallet and $D_H = 14$ m while dead heading. Meaning the high lift travelled a total of 31 m to complete the entire job list.

A.1.3 Single pallet with same side queueing

The first step is calculating the total distance travelled while a high lift is occupied, carrying a pallet. Substituting the relevant information equations (A.1) and (A.2) are obtained.

$$D_P = \sum_{i=1}^6 [x_i + y_i - 1] \quad (\text{A.3})$$

$$D_H = \sum_{i=1}^2 [x_i + y_i - 1] + \sum_{i=5}^2 [x_i + y_i - 1] + |x_I - x_{I+1}| + |y_I - y_{I+1}| \quad (\text{A.4})$$

Table A.3 summarises the distance a high lift moved to complete each job in the generated job list example. It is important to note that the calculation to determine the distance dead headed (D_H) when storing pallet 3 and retrieving pallet 4 are simultaneous.

i	Movement type	x	y	$D_P(A.3)$	$D_H(A.4)$
1	Storage	6	3	$6 + 3 - 1 = 8$	$6 + 3 - 1 = 8$
2	Storage	6	2	$6 + 2 - 1 = 7$	$6 + 2 - 1 = 7$
3	Storage	5	3	$5 + 3 - 1 = 7$	$ 5-1 + 3-3 = 4$
4	Retrieval	1	3	$1 + 3 - 1 = 3$	
5	Retrieval	2	1	$2 + 1 - 1 = 2$	$2 + 1 - 1 = 2$
6	Retrieval	2	2	$2 + 2 - 1 = 3$	$2 + 2 - 1 = 3$

Table A.3: Summary of the results obtained when substituting the generate job list example into the distance equation.

From the calculated distances summarised in Table A.3 it is possible to find the total occupied and dead heading distances the high lift travelled. Summing over the distances the high lift travelled a total of $D_P = 30$ m with a pallet and $D_H = 24$ m while dead heading. Meaning the high lift travelled a total of 54 m to complete the entire job list.

A.1.4 Paired pallet movement logics

The job list in Table A.1 is modified into the job list shown in Table A.4. Since this job list is used by paired pallet movement logics one pallet is stored and another retrieved in one job.

i	Movement type	x	y
1	Storage	6	3
2	Retrieval	1	3
3	Storage	6	2
4	Retrieval	2	1
5	Storage	5	3
6	Retrieval	2	2

Table A.4: Generated job list for the movement logics that moves one pallet per job.

In the modified example there are still three storage pallets ($I = 3$) and three retrieval pallets ($O = 3$). However, since a paired pallet movement logics is used there are only three jobs in the job list ($K = 6$) Using this example the total movement distance a high lift travels is calculated for three of the movement logics.

A.1.5 Paired jobs with same side queuing

Substituting the relevant information equations (A.5) and (A.6) are obtained.

$$D_P = \sum_{i=1}^6 [x_i + y_i - 1] \quad (A.5)$$

$$D_H = \sum_{i=1} |x_i - x_{i+1}| + |y_i - y_{i+1}| \quad i = 1, 3, 5. \quad (A.6)$$

Table A.5 summarises the distance a high lift moved to complete each job in the generated job list example. It is important to note that the calculation to determine the distance dead headed (D_H) are simultaneous.

i	Movement type	x	y	$D_P(A.5)$	$D_H(A.6)$
1	Storage	6	3	$6 + 3 - 1 = 8$	$ 6-1 + 3-3 = 5$
2	Retrieval	1	3	$1 + 3 - 1 = 3$	
3	Storage	6	2	$6 + 2 - 1 = 7$	$ 6-2 + 2-1 = 5$
4	Retrieval	2	1	$2 + 1 - 1 = 2$	
5	Storage	5	3	$5 + 3 - 1 = 7$	$ 5-2 + 3-2 = 4$
6	Retrieval	2	2	$2 + 2 - 1 = 3$	

Table A.5: Summary of the results obtained when substituting the generate job list example into the distance equation.

From the calculated distances summarised in Table A.5 it is possible to find the total occupied and dead heading distances the high lift travelled. Summing over the distances the high lift travelled a total of $D_P = 30$ m with a pallet and $D_H = 14$ m while dead heading. Meaning the high lift travelled a total of 44 m to complete the entire job list.

A.1.6 Paired pallets with opposite side queuing

Substituting the relevant information equations (A.5) and (A.6) are obtained.

$$D_P = \sum_{i=1} [6 - x_i + y_i + x_{i+1} + y_{i+1} - 1] \quad i = 1, 3, 5. \quad (A.7)$$

$$D_H = \sum_{i=1} [|x_i - x_{i+1}| + |y_i - y_{i+1}| + 7] \quad i = 1, 3, 5. \quad (A.8)$$

i	Movement type	x	y	$D_P(A.7)$	$D_H(A.8)$
1	Storage	6	3	$6 - 6 + 3 + 1 + 3 - 1 = 6$	$ 6-1 + 3-3 + 7 = 12$
2	Retrieval	1	3		
3	Storage	6	2	$6 - 6 + 2 + 2 + 1 - 1 = 4$	$ 6-2 + 2-1 + 7 = 12$
4	Retrieval	2	1		
5	Storage	5	3	$6 - 5 + 3 + 2 + 2 - 1 = 7$	$ 5-3 + 3-2 + 7 = 11$
6	Retrieval	2	2		

Table A.6: Summary of the results obtained when substituting the generate job list example into the distance equation.

From the calculated distances summarised in Table A.6 it is possible to find the total occupied and dead heading distances the high lift travelled. Summing over the distances the high lift travelled a total of $D_P = 17$ m with a pallet and $D_H = 35$ m while dead heading. Meaning the high lift travelled a total of 52 m to complete the entire job list.

A.1.7 Paired pallets with cyclic queueing

Substituting the relevant information equations (A.9), (A.10) and (A.11) are obtained.

$$D_{P1} = \sum_{i=3} [5 - x_i + y_i + x_{i+1} + y_{i+1}] \quad i = 1, 5. \quad (\text{A.9})$$

$$D_{P2} = \sum_{i=1} [5 + x_i + y_i - x_{i+1} + y_{i+1}] \quad i = 3 \quad (\text{A.10})$$

$$D_H = \sum_{i=1} |x_i - x_{i+1}| + |y_i - y_{i+1}| \quad i = 1, 3, 5. \quad (\text{A.11})$$

Table A.5 summarises the distance a high lift moved to complete each job in the generated job list example. It is important to note that the calculation to determine the distance dead headed (D_H) are simultaneous.

i	Type	x	y	$D_{P1}(\text{A.9})$	$D_{P2}(\text{A.10})$	$D_H(\text{A.11})$
1	Storage	6	3	$5 - 6 + 3 + 1 + 3 = 6$	$5 + 6 + 2 - 2 + 1 = 12$	$ 6-1 + 3-3 = 5$
2	Retrieval	1	3			
3	Storage	6	2			$ 6-2 + 2-1 = 5$
4	Retrieval	2	1	$5 - 5 + 3 + 2 + 2 = 7$		
5	Storage	5	3			
6	Retrieval	2	2			$ 5-2 + 3-2 = 4$

Table A.7: Summary of the results obtained when substituting the generate job list example into the distance equation.

From the calculated distances summarised in Table A.6 it is possible to find the total occupied and dead heading distances the high lift travelled. Summing over the distances the high lift travelled a total of $D_{P1} = 13$ m and $D_{P2} = 12$ m with a pallet and $D_H = 14$ m while dead heading. Meaning the high lift travelled a total of 34 m to complete the entire job list.

A.2 Tabu search meta heuristics: Worst pairing Tabu search

Due to the similarities between the **Pairing from the best** and **Pairing from the worst** Tabu search algorithms the pseudocode for the **Pairing from the worst** Tabu search is shown below, see Algorithm (5). This algorithm follows the same flow as Algorithm (4) with two minor changes. The first change is in line 3 where the distance of the current worst job pairing is initialised with a value of negative infinity. The next change is in line 13 where a new job parining candidate is only accepted if it worse that than the current worst job pairing.

Algorithm 5: Worst pairing Tabu search**Input** : List of jobs in any sequence and movement type**Output:** Good sequenced list of jobs.

```

1   $s \leftarrow s_0$ ;
2   $s_{\text{Best}} \leftarrow s$ ;
3   $w_{\text{Worst}} \leftarrow -\infty$ ;
4   $\text{tabuList} \leftarrow \text{null}$ ;
5   $\text{worstTabuList} \leftarrow \text{null}$ ;
6  while (not stoppingCondition()) do
7      for  $i < \text{length}(s)-1$  do
8           $w_{\text{Candidate1}} \leftarrow s_i$ ;
9           $w_{\text{Candidate2}} \leftarrow s_{i+1}$ ;
10         if  $\text{jobType}(w_{\text{Candidate1}}) = \text{Storage}$  and  $\text{jobType}(w_{\text{Candidate2}}) = \text{Retrieval}$  then
11              $w_{\text{Candidate}} \leftarrow [w_{\text{Candidate1}}, w_{\text{Candidate2}}]$ ;
12             if (not containsTabuElements(Candidate, worstTabuList)) then
13                 if  $\text{fitness}(\text{tempSequence}) > w_{\text{Worst}}$  then
14                      $\text{Candidate1} \leftarrow w_{\text{Candidate1}}$ ;
15                      $\text{Candidate2} \leftarrow w_{\text{Candidate2}}$ ;
16                      $\text{worstTabuList} \leftarrow \text{addElement}(w_{\text{Candidate}})$ ;
17                     while ( $\text{size}(\text{worstTabuList}) > \text{maxTabuListSize}$ ) do
18                          $\text{removeElement}(\text{worstTabuList})$ 
19                     end
20                 end
21             end
22         end
23     end
24      $\text{Candidate1} \leftarrow \text{mostFrequent}(\text{Candidate1}, \text{Candidate2})$ ;
25     for ( $j < \text{length}(s)$ ) do
26          $\text{candidateList} \leftarrow \text{null}$ ;
27          $\text{Candidate} \leftarrow \text{null}$ ;
28          $\text{Candidate2} \leftarrow s_j$ ;
29         if ( $\text{Candidate1} \neq \text{Candidate2}$  and  $\text{mostCommon}(\text{Candidate2})$ ) then
30              $\text{Candidate} \leftarrow \text{Candidate} + \text{Candidate1} + \text{Candidate2}$ ;
31             if (not containsTabuElements(Candidate, tabuList)) then
32                  $\text{candidateList} \leftarrow \text{candidateList} + \text{Candidate}$ ;
33             end
34         end
35          $\text{testFitness}(s_{\text{best}}, \text{candidateList}, \text{tabuList})$ 
36     end
37 end

```

APPENDIX B

Tables and graphs

B.1 Data

This chapter contains tables and graphs regarding the statistical techniques used in Chapter 3.

B.1.1 Expected travel times of the high lifts

Using the time equation $T = (P_W \times C)/V_h + (P_H \times L)/V_v$ as discussed in § 3.4.2 the time, in seconds, is calculated to between each rack location and the queue. The results are summarised in Figure B.1 for each level of rack locations respectively. In this instance the high lift's vertical velocities are equal irrespective of the type of the movement type.

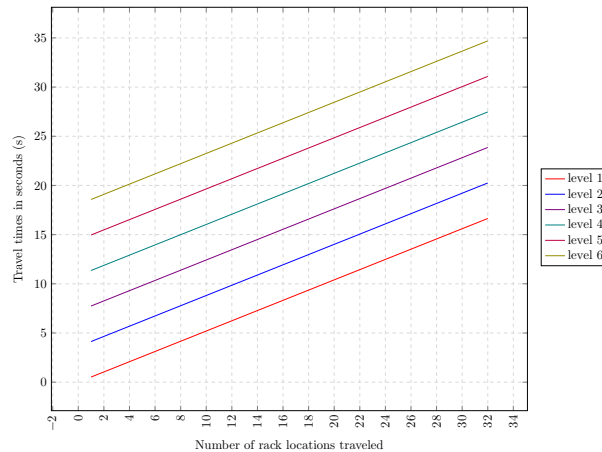


Figure B.1: Graphic representation of the travel times to store a pallet. The times are generated using specifications for a EKX 410 Electric order picker/tri-lateral stacker.

B.1.2 Regression models

The historical data is divided into two data sets. The first set contains all storage times stamps extracted from the historical data while the second set contains all retrieval times stamps extracted from the historical data. The data from these sets are then plotted separately as shown in Figure 3.1 in § 3.4.2. All anomalies are removed from the two data sets and the sets are plotted as shown in Figure 3.3(b) in § 3.4.2.

Linear regression lines are added the four data sets. Using the linear regression lines equations capable of determining each line's correlation coefficient and are summarised in Table B.1.

	Phase 1 storage times	Phase 1 retrieval times	Clean storage times	Clean retrieval times
level 1	$-1.349x + 16.45$	$0.319x + 31.18$	$-1.094x + 7315$	$0.433x + 36.52$
level 2	$0.592x + 94.97$	$0.650x + 36.15$	$0.603x + 50.01$	$0.667x + 33.75$
level 3	$0.593x + 98.93$	$0.460x + 37.79$	$0.726x + 53.44$	$0.597x + 35.99$
level 4	$0.145x + 91.66$	$0.473x + 41.44$	$0.711x + 59.33$	$0.667x + 33.75$
level 5	$0.758x + 108.8$	$0.193x + 46.71$	$0.798x + 64.97$	$0.483x + 43.72$
level 6	$0.410x + 102.8$	$0.217x + 51.71$	$0.811x + 72.76$	$0.493x + 49.39$

Table B.1: A table summarising the correlation coefficient equations for each linear regression line of the data sets. The **clean** columns refer to the data sets that have been cleaned of all anomalies.

Using the method described in § 3.5.1 the different stages of the generated travel times' distribution is showed. As more travel times are generated the number of generated data point exceed the number of historical data points. To properly compare the histogram of the generated data to the historical data the data sets need to be standardised, using the **STANDARDIZE** function in Microsoft Excel. The standardised data sets are summarised in Table B.2.

Time	Entries per time interval					Standardised entries per time interval				
	Historical	Stage 1	Stage 2	Stage 3	Stage 4	Historical	Stage 1	Stage 2	Stage 3	Stage 4
20	0	0	0	0	0	—	-0.95	-0.84	-0.95	-0.96
21	0	0	0	0	0	—	-0.95	-0.84	-0.95	-0.96
22	0	0	0	0	0	—	-0.95	-0.84	-0.95	-0.96
23	1	0	14	146	1445	—	-0.92	-0.84	-0.92	-0.92
24	4	0	75	600	5671	—	-0.81	-0.84	-0.77	-0.81
25	5	0	67	749	7374	—	-0.78	-0.84	-0.79	-0.77
26	17	3	267	2483	25129	—	-0.36	-0.21	-0.32	-0.36
27	22	3	306	3271	32352	—	-0.18	-0.21	-0.22	-0.17
28	20	5	269	2955	29454	—	-0.25	0.21	-0.31	-0.25
29	33	4	450	4815	48575	—	0.20	0.00	0.12	0.19
30	47	12	689	6906	69225	—	0.70	1.68	0.69	0.69
31	66	7	974	9694	96773	—	1.36	0.63	1.36	1.36
32	89	14	1273	13241	130620	—	2.17	2.10	2.07	2.20
33	87	9	1340	12928	128397	—	2.10	1.05	2.23	2.13
34	85	16	1265	12349	124815	—	2.03	2.53	2.06	1.99
35	50	10	720	7212	73464	—	0.80	1.26	0.76	0.77
36	37	4	567	5399	53970	—	0.34	0.00	0.40	0.33
37	36	3	479	5403	53225	—	0.31	-0.21	0.19	0.33
38	26	2	397	3712	38264	—	-0.04	-0.42	-0.01	-0.07
39	12	5	190	1804	17872	—	-0.53	0.21	-0.50	-0.52
40	11	1	169	1684	16022	—	-0.57	-0.63	-0.55	-0.55
41	8	1	122	1194	11825	—	-0.67	-0.63	-0.66	-0.67
42	7	1	93	991	10358	—	-0.71	-0.63	-0.73	-0.72
43	12	0	194	1764	17759	—	-0.53	-0.84	-0.49	-0.53
44	5	0	80	700	7411	—	-0.78	-0.84	-0.76	-0.79

Table B.2: Standardised data sets.

Table B.2 summarises the data from the historical histogram as well as four stages of the generated travel time histogram. The different stages refers to the number of travel times, in seconds, that have been generated and placed into their corresponding bins — 100, 1000, 10000, 1000000 receptively.

Figure B.2 visually compares the different stages of the estimated travel times to that of the historical data, in terms of the standardised data. It shows that as more travel times are generated the data distribution start looking similar to the historical data's distribution. When

Stage 4 is reached the generated and historical graphs look almost identical.

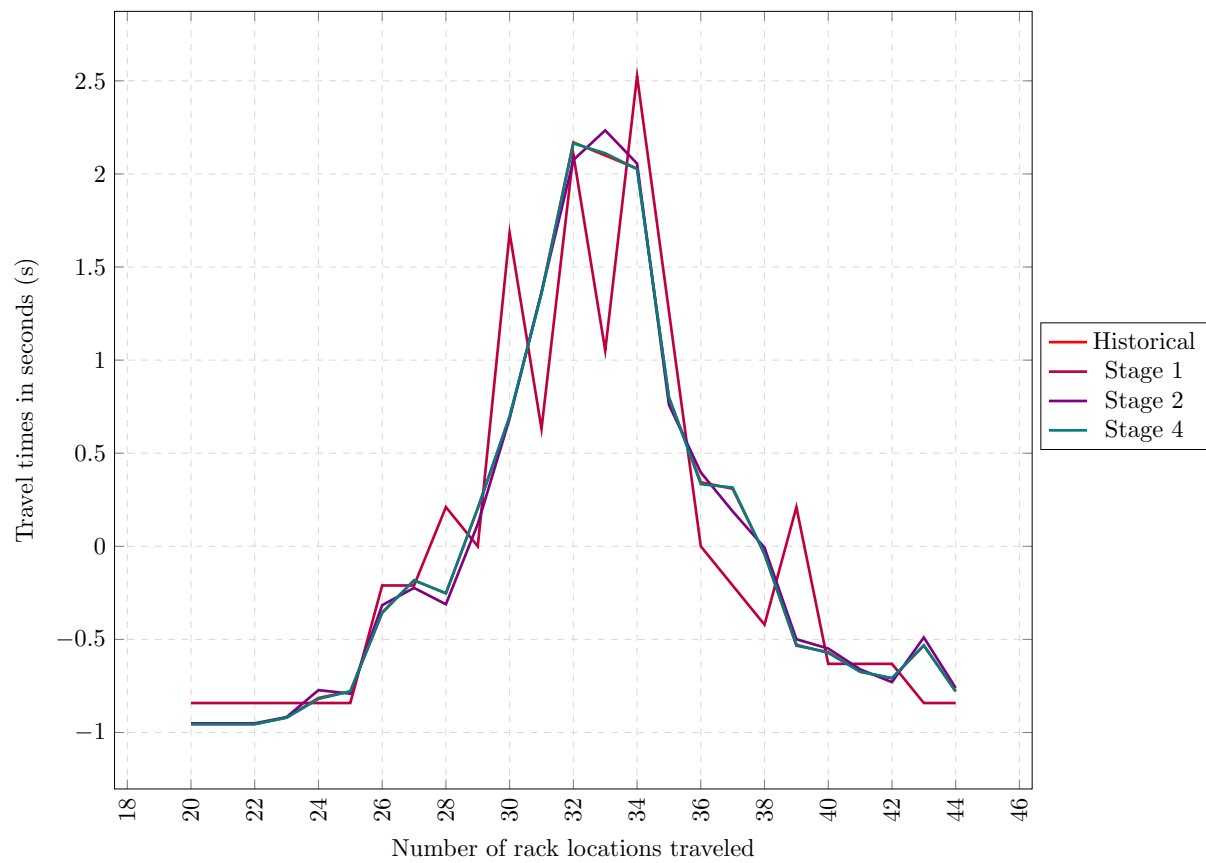


Figure B.2: Visual standardised comparison between the historical data and the estimated data.

B.1.3 Relocation travel times

Table B.3 summarises the linear and multivariate equations investigated to possible estimate travel times, in seconds. The three tables contains the equations used for the three different parameters of the triangle distribution. Table B.3 shows that none of the regression equations can be used to estimate the minimum or maximum value parameters of the triangle distribution. However, when determining the value that resembles the expected travel time value a multivariate equation fits the historical data tightly insinuating that further investigation should be done.

Minimum travel times				
	Linear equation	R^2	Multivariate equation	R^2
Level 1	21	N/A	$0.01x_h + 0.23x_v + 20.57$	0.18
Level 2	21.04	0.002	$0.01x_h + 0.23x_v + 20.57$	0.18
Level 3	$0.008x_h + 20.95$	0.068	$0.01x_h + 0.23x_v + 20.57$	0.18
Level 4	21.23	0	$0.01x_h + 0.23x_v + 20.57$	0.18
Level 5	$-0.001x_h + 21.12$	0.002	$0.01x_h + 0.23x_v + 20.57$	0.18
Level 6	$0.057x_h + 21.60$	0.099	$0.01x_h + 0.23x_v + 20.57$	0.18

Average travel times				
	Linear equation	R^2	Multivariate equation	R^2
Level 1	$-0.013x_h + 51.92$	0.003	$0.77x_h + 5.07x_v + 41.59$	0.84
Level 2	$0.645x_h + 49.47$	0.96	$0.77x_h + 5.07x_v + 41.59$	0.84
Level 3	$0.738x_h + 51.83$	0.82	$0.77x_h + 5.07x_v + 41.59$	0.84
Level 4	$0.756x_h + 56.21$	0.75	$0.77x_h + 5.07x_v + 41.59$	0.84
Level 5	$0.855x_h + 60.75$	0.67	$0.77x_h + 5.07x_v + 41.59$	0.84
Level 6	$0.869x_h + 65.67$	0.67	$0.77x_h + 5.07x_v + 41.59$	0.84

Maximum travel times				
	Linear equation	R^2	Multivariate equation	R^2
Level 1	$-1.373x_h + 157.1$	0.4	$1.57x_h + 7.03x_v + 94.91$	0.63
Level 2	$1.478x_h + 103.1$	0.79	$1.57x_h + 7.03x_v + 94.91$	0.63
Level 3	$1.597x_h + 108$	0.63	$1.57x_h + 7.03x_v + 94.91$	0.63
Level 4	$1.616x_h + 114.8$	0.61	$1.57x_h + 7.03x_v + 94.91$	0.63
Level 5	$1.77x_h + 121.5$	0.56	$1.57x_h + 7.03x_v + 94.91$	0.63
Level 6	$1.796x_h + 123.7$	0.45	$1.57x_h + 7.03x_v + 94.91$	0.63

Table B.3: Summary of the equations obtained when the linear and multivariate regression models are applied. The models are applied to the combined data sets for the minimum, average and maximum travel times. The correlation coefficients are also listed for each level.

Table B.4 summarises the minimum, median and maximum of the historical travel times per rack location. Due to the size of the table and number of data points the table is placed here.

Table B.4: Summary of the maximum, minimum and median travel times, in seconds, to each rack location.

B.1.4 Job list generation

Table B.5 summarises a break down of anomalies found in the historical data of November 2015. Anomaly 1 are jobs that have duplicate starting and ending locations. Anomaly 2 are jobs with where the same pallet is stored from different locations. Anomaly 3 are when pallets are stored in already occupied rack locations. Anomaly 4 represents the cases where a job has exact duplicates in the data set. Table B.5 also shows the number of jobs performed, per day, of a specific movement type.

Day	Total jobs	Storage	Retrieval	Movement	Anomaly 1	Anomaly 2	Anomaly 3	Anomaly 4	CU%
1	0	0	0	0	0	0	0	0	0%
2	1307	914	378	15	15	584	5	52	50%
3	1102	714	378	10	10	410	25	33	43%
4	1569	1049	507	13	13	621	28	74	47%
5	1475	1065	398	12	12	648	41	67	52%
6	1097	850	237	10	10	500	26	76	56%
7	1375	1306	65	4	4	764	3	39	59%
8	0	0	0	0	0	0	0	0	0%
9	1563	1309	246	8	8	802	10	46	55%
10	1436	1023	389	24	24	626	22	79	52%
11	2003	1421	554	28	28	911	40	77	53%
12	1479	1186	286	7	7	653	24	41	49%
13	2149	1613	513	23	23	976	66	17	50%
14	1653	1287	357	9	9	747	36	33	50%
15	0	0	0	0	0	0	0	0	0%
16	2089	1753	317	19	19	1028	38	35	54%
17	2542	2099	426	17	17	1290	40	69	56%
18	2095	1689	386	20	20	986	74	66	55%
19	2291	1710	568	13	13	1078	32	24	50%
20	1483	1191	284	8	8	663	64	116	57%
21	1311	1075	229	7	7	643	54	25	56%
22	0	0	0	0	0	0	0	0	0%
23	1846	1419	417	10	10	907	40	83	56%
24	1861	1222	623	16	16	832	27	66	51%
25	1844	1394	436	14	14	843	23	43	50%
26	1555	1108	427	20	20	634	18	40	46%
27	1312	1051	254	7	7	611	34	69	55%
28	1506	1159	338	9	9	688	30	34	51%
29	0	0	0	0	0	0	0	0	0%
30	2334	2120	198	16	16	1395	16	171	68%

Table B.5: A table representing the number of jobs completed each day including the number of anomalies for November 2015.

APPENDIX C

Decision support system

Companies are often cautious to invest time and money into new developments. This is generally caused by their lack of ability to analyse the effect of the new development [15]. To aid companies faced with this predicament decision support system (DSS) are developed to assist them in making a decision regarding new developments. A decision support system is a computerized information system used to support decision making. A DSS allows the users to process, analyse large amounts of data and gather information that can be used to assist the user to solve problems and make better decisions.

The simulation model used to solve the scope of this study is accompanied by a DSS to support in the understanding of each algorithm and logic. At start-up the user is presented with the main menu of the simulation model, shown in Figure C.1. Before the simulation is able to start the user is required to select the combination of algorithms, logics, data, number of high lifts and a simulation time modifier from the parameter settings. Selecting the movement logic **PEP** enables the real world scenario observed at PEP's DC and uses the built in selection for this logic. Because this selection utilises built in parameter settings the Sequencer algorithm, Placement algorithm, Data selection and High lift selection criteria becomes disabled.

If **PEP** is not selected a high lift logic and placement algorithm needs to be selected. It is also required that the sequencer is either set to none or enabled. Included under the **Placement** selection is **Custom** that allows the user to load any placement configuration not included as a default in the DSS. Using the a file browser a placement configuration can easily be loaded into the DSS from anywhere on the computer. Further the user is required to select between the three options underneath the **Data** parameter, namely Historical, Generated and Load. The **Historical** parameter runs the simulation using job lists constructed from PEP's historical data as described in Chapter 3. The **Generated** parameter creates a job lists when the simulation models starts based on setting defined by the user. Selecting the load parameter allows the user to load a previously created job list using the file browser. The number of high lifts can be set to a value between 1 and 4 and regulates the number of high lifts that are constantly working in rack system S1. Changing the speed multiplier changes the rate at which the simulated time is updated per second. A speed multiplier of 1 indicates that the simulated time is updated once every second and can be increased to a maximum speed of 500 updates per second. Figure C.2 shows an example of the settings the users can manipulate when **Generated** data is selected.

The user is able to specify the total number of jobs that need to be simulated. It is important to note that due to the size of the storage rack system, 192 rack locations and 46 racks, a maximum of 8832 pallets can be stored in the storage racks. These jobs are divided among the selected number of shift as indicated. The shift variation determines how even the jobs are divided

The screenshot shows a software window titled "Parameters:" and "Simulation:". The "Parameters:" section contains five groups of radio buttons: "HighLift Logic" (PEP, SSP, OSP, OS, SS, CS), "Sequencer" (None, Enabled), "Placement" (Columns, Rows, Steps, XY Distance, Pythagoras, Custom), "Data" (Historical, Generated, Load), and "Number of Highlifts" (a dropdown menu set to 1). Below these is "Speed Multiplier" (a dropdown menu set to 1). The "Simulation:" section includes labels for "Total Jobs:", "Jobs Completed:", "Distance Moved:", and "Simulation timer:". A "Total Progress:" label is positioned above a horizontal progress bar. At the bottom center is a "Start" button.

Figure C.1: The main menu of the simulation model's decision support system.

The screenshot shows a "Data Generator Settings" window. It contains several input fields and sliders: "Number of jobs:" (text box with 5200), "Number of shifts:" (text box with 5), "Shift variation(%):" (slider set to 25), "Job ratio: (Storage | Retrieval)" (slider set to 50), "Start date:" (three dropdown menus showing 2015, Aug, 27), "Start time:" (three dropdown menus showing 11, 19, 0), and "Initial occupied cells(%):" (slider set to 15). An "Accept" button is located at the bottom.

Figure C.2: Visualisation of the settings menu when data is generated for the simulation model.

amongst the shifts as well as aisles. A shift variation percentage of zero means the jobs are divided equally and as the percentage increases the number of jobs per shift and aisle becomes more random. The job ratio determines the storage to retrieval job ratio and is by default set to 50/50. A ratio of 0/100 indicated that no storage jobs are performed and 100/0 means to retrieval jobs are performed. The job ratio is applied to each aisle individually. The user is also able to set the start time and date to increase the aesthetic and readability if the generated data. Lastly, the user may set the percentage of static pallets in the initial rack system. Static pallets are pallets that are not stored or retrieved during the indicated time interval and is allocated to rack location at random at the beginning of the simulation model. Once the user selected all the settings the **Accept** button saves the settings until it is used when the simulation model starts. This feature is not intended to be used when data needs to be simulated over a very long period. Its usefulness rather lies at quickly simulating a data set to investigate different parameters.

When all the parameters are set according to the user's preference the simulation model is executed by pressing the **Start** button. Figure C.3 shows an example of the DSS while it is simulating the movement of high lifts using the PEP movement logic parameter. At the start of the simulation model the DSS reads the total number of jobs in the job list. As jobs are completed in the simulation model the DSS is updated in real time. The DSS displays the total number of jobs completed, total cumulative distance moved by high lifts and the simulated time passed. As an added visual aid the total progress of the simulation is displayed in a progress bar.

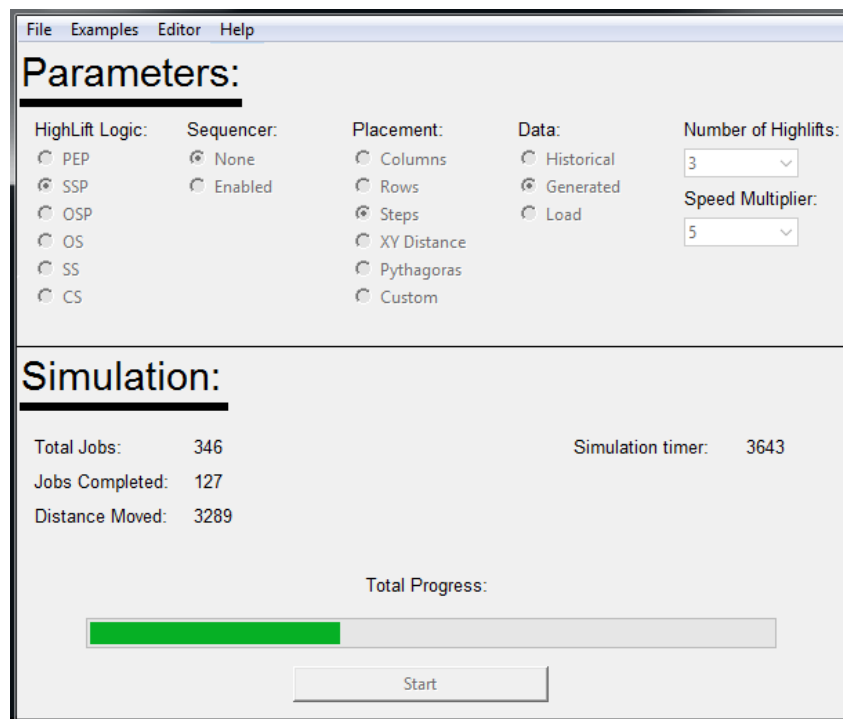


Figure C.3: The main menu of the simulation model's decision support system after the simulation is started.

By giving the user to ability to change the parameters and displaying the results of the simulation might not always be enough to persuade the user into making a decision. For this reason multiple additional aids are added and can be accessed through the menu bar.

The **Examples** tab expands to display examples tabs for high lift logics, Sequencer algorithm and placement algorithms. Each of these tabs can further be expanded allowing the user to

select between a visualisation or simulation of each logic and algorithm. Figure C.4 shows the DSS when the visualisation of movement logics is selected from the **Examples** tabs.

The visualisation option of movement logics is an animation of how the movement logics are performed. At the bottom of the screen two buttons allow the user the cycle through the different movement logics. By cycling through the movement logics the name and description of each is displayed at to top of the screen. The animation shows the storage and retrieval of pallets according with the “L” representing the queue closest to the picking area and “R” the queue between the two rack systems. The total movement distance and dead heading is displayed to give the user a better understanding of how each logic performs.

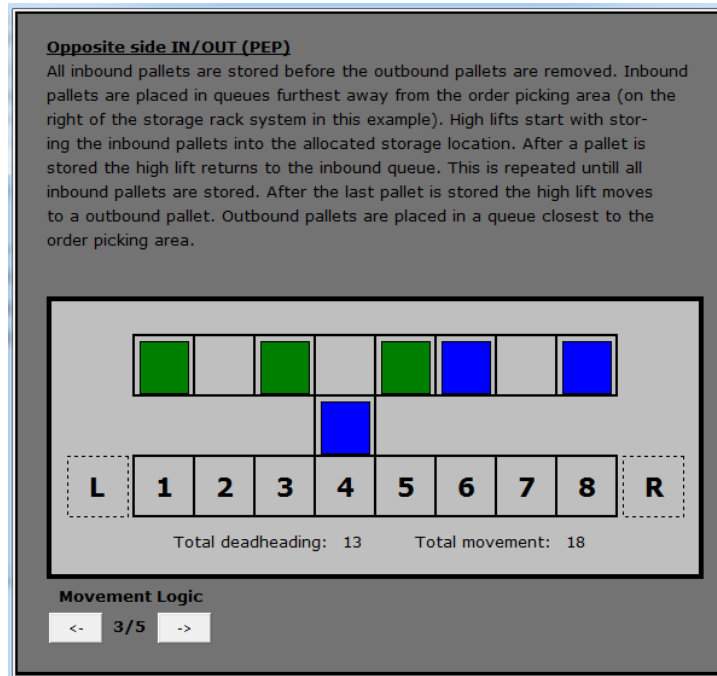


Figure C.4: A visualisation of the different movement logics under the **Examples** tab.

When the simulation option is selected the user is presented with a window as shown in Figure C.5. The simulation option allows the user to cycle through the different movement logics using buttons at the bottom of the screen and displays the logic name and description at the top of the screen. The animation’s default speed is sped up to 500 times the actual time but can temporally be reverted to normal speed by clicking the **Speed** button. After each job list, consisting of 18 jobs, is completed the total distance, total dead heading and total movement distance is recorded in the text box with moving averages at the bottom. Every time a logic is changed the simulation animation is restarted and all stored data removed.

When selecting the visualisation for either the **Placement** or **Sequencer** from the examples tab a window is presented as shown in Figure C.6. The visualisation for these two work the same as previously explained except a job list is present for each. As the animation completes jobs the arrow in the job list updates making it clear which job is currently being done. It is possible to cycle through both the movement logics and placement algorithms. Every time a logic/algorithm is changed the job list’s layout is changed according to the selection.

An further aid that is included is the **Editor** that helps the user to get a better understanding of how each combination of the different movement logics, placement algorithms and a sequencer has on the system. Utilising the editor the user has full control over the number of storage and retrieval job present as well as the logic and algorithms used.

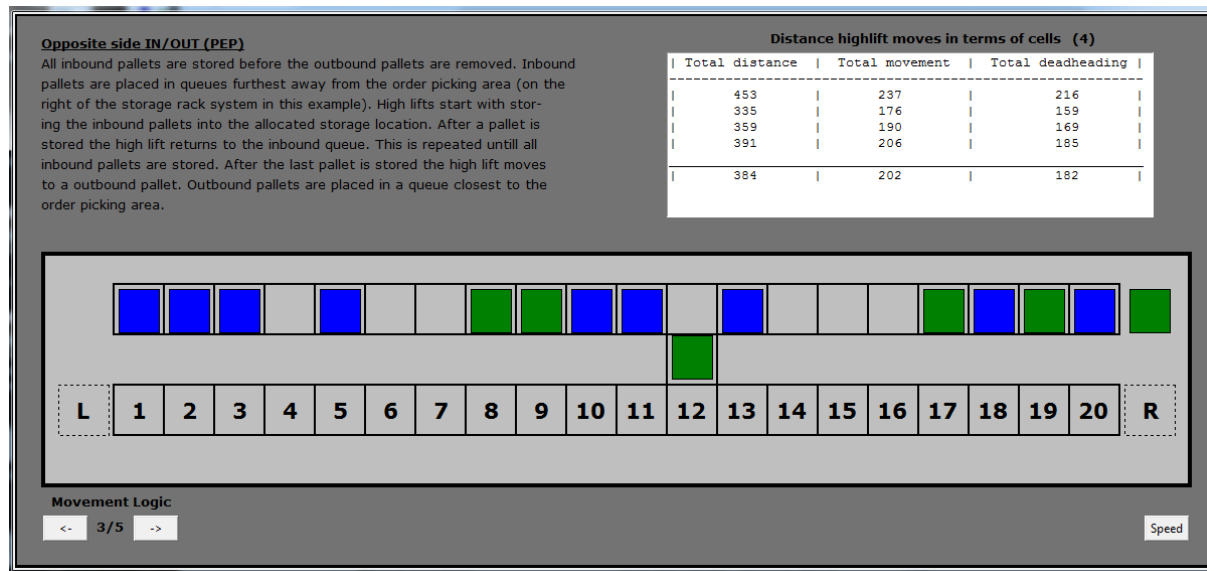


Figure C.5: A visualisation of simulating different movement logics under the "Examples" tab.

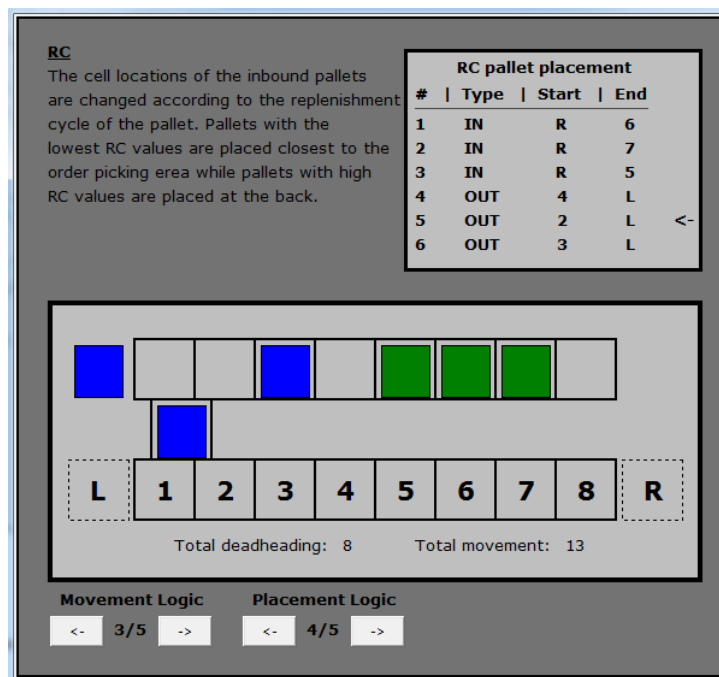


Figure C.6: A visualisation of the placement algorithms under the "Examples" tabs.

The first tab underneath **Editor** is the rack painter shown in Figure C.7. The rack painter allows the user to specify the replenishment cycle regions by painting them onto the canvas. The painter is capable of assigning RC1–RC4 as well as the reserve areas to the rack system by using a paint brush. Each region is assigned an unique colour, shown in the block numbered 2, where RC1 = Green, RC2 = Yellow, RC3 = Blue, RC4 = Red, R1 = Line, R2 = Orange, R3 = Cyan. Also included is an eraser to clear any unwanted regions. After a colour is selected there is three different approaches to apply them to the canvas.

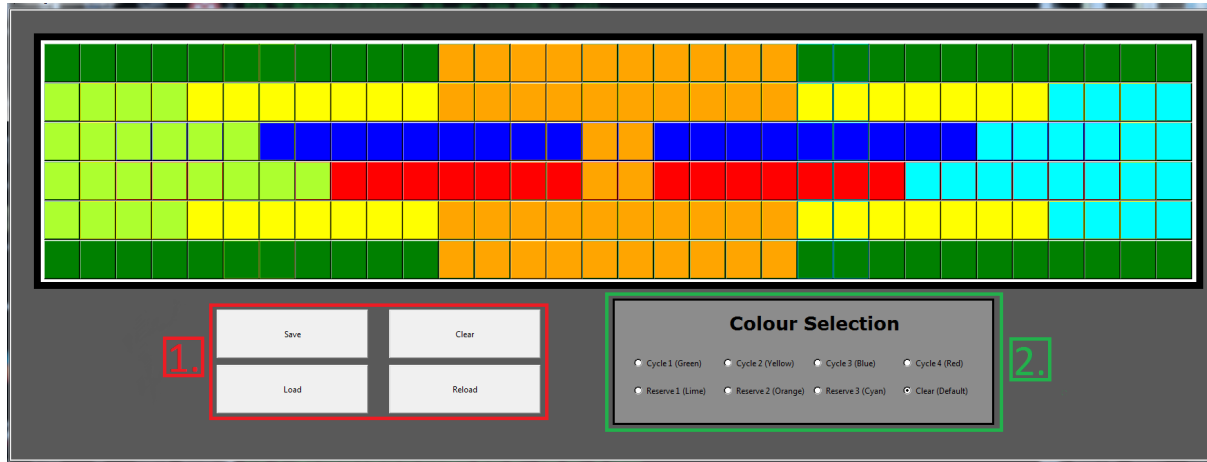


Figure C.7: A visualisation of the rack painter underneath the "Editor" tab.

The first is to simply left click on the rack location that the colour needs to be applied to. This will only change the selected rack location to the desired colour. The next method is to double click the left mouse button anywhere on the canvas. This activates the brush tool for the specified colour allowing the user to apply the colour to multiple rack locations. While the brush tool is activated any rack location the mouse pointer moves over is changed to the selected colour. The last method is activating the brush tool by pressing the right mouse button anywhere on the canvas. By clicking, not double clicking, anywhere on the canvas the brush tool is deactivated. Two methods of activating the brush tool is included as the DSS is very sensitive when it comes to the time difference between clicks performed to determine if it was a double click or just two single clicks. Thus if the user finds that double clicks are too often interpreted as single clicks, using the right mouse button instead helps reduce frustration. Only clicks on the canvas is registered to decrease the possibility of accidentally deactivating the brush tool while painting. While the brush tool is activated all buttons on the screen become deactivated to prevent accidentally pressing one.

The buttons in the box marked with "1." allow the user to quickly alter the rack configuration. When the **Save** button is pressed a file browser is opened allowing a csv file containing the configuration to be saved wherever the user sees fit. In the case that the configuration contains rack locations that do not have a colour assigned to it an error message appears. The error message gives the user a choice of either cancelling the attempt to save or allowing the save to happen. If the save is allowed all unassigned rack locations are automatically assigned as a R1 rack location.

Using the load button opens the file browser allowing the user to navigate to a previously saved rack configuration. When a rack configuration is loaded the current configuration is discarded and the canvas is filled with the loaded colours. The **Clear** button removes all the assigned colours and reverts the rack location back to blank. This is handy when an entire configuration must quickly be erased. The **Reload** button is mainly used in unison with a loaded configuration. After changes are made to a loaded configuration the user is able to quickly reset the canvas to

the original configuration that was loaded. This allows the user to save time by removing the entire process of reloading the configuration using the file browser. If no file is loaded prior to pressing the **Reload** button a clear configuration is loaded.

The next aid underneath the **Editor** tabs is the rack visualiser. The rack visualiser generates an job list according to user specifications regarding storage jobs, retrieval jobs and occupied rack locations. Using the generated job list its current configuration is compared to if it used a different movement logic, placement algorithm and sequencer as shown in Figure C.8

The top rack system displays the configuration of the generated job list. The bottom rack system shows the configuration of the same job list after a logic and algorithms are applied. The grey rack locations represent the static pallets in the storage rack. As these are not handled in the observed time interval they are in the same locations for both systems. The red rack locations represent retrieval jobs and is in the same location for both systems as they are already in the system before the logics/algorithms are applied. Green rack locations indicate the storage jobs and is on different location in each system as the logics/algorithms are applied to them in the bottom system.

The number in each rack location represents the RC value assigned to it. Both systems uses the same RC configuration to allow for an accurate comparison, however, the top system does not use it while storing pallets. Also indicated on the screen are the distances calculated in rack locations for each system. From the example in Figure C.8 it shows that the bottom system performs far better than the top one with the selected parameters.

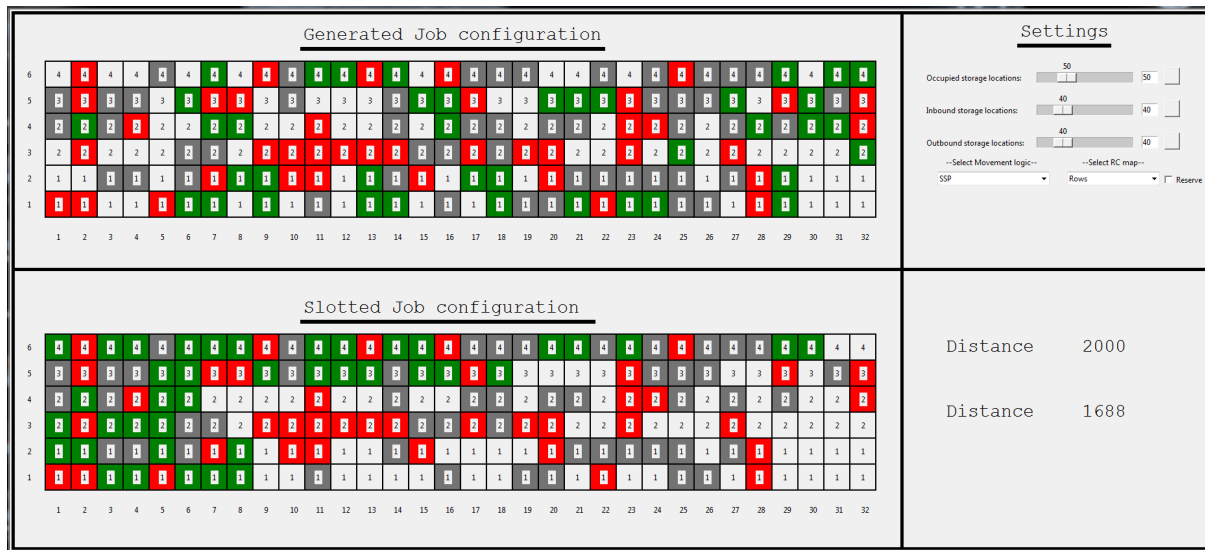


Figure C.8: A visualisation of the rack display underneath the **Editor** tab.

The parameters are selected under the setting for the window as depicted by Figure C.9. Using the sliders the number of pallets of each type is selected. As there is only 192 rack locations the maximum amount of pallets that can be assigned is 192. Each slider is able to go to a maximum of 192 but if the total number of assigned pallets reaches 192 the sliders are locked. The sliders only unlocks when the total number of assigned pallets are less than 192. slider as well as in the systems.

The number of pallets can be set in three ways. The first method is by sliding the value selector on the track. This method is fast but not very accurate. For a more accurate method it is possible to click directly on the track on the side in which the value should move. Each click moves the value by one and enables the user to precisely set the parameter to the desired value.

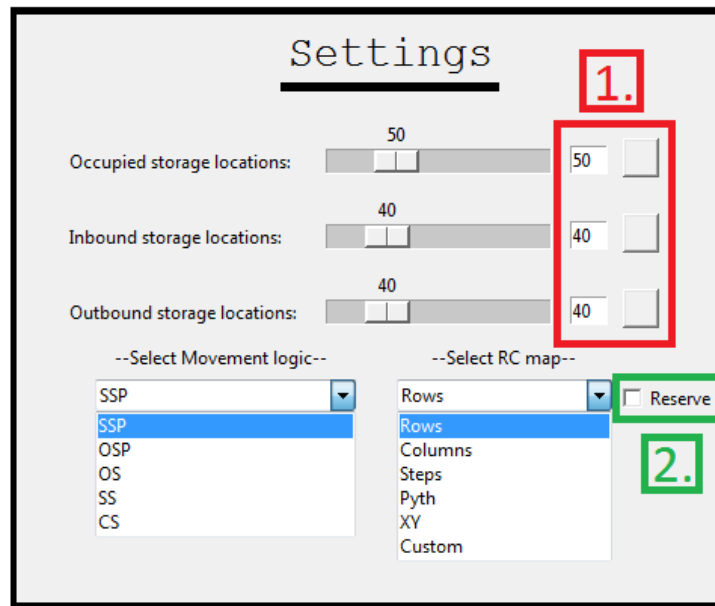


Figure C.9: A visualisation of the settings available when using the rack display GUI.

The last method is using the text insert box, indicated by the box numbered 1, by typing the value in the empty box and pressing the button next to it.

The movement logic list box contains all the movement logics described in Chapter 2 and allow for the logic parameter to be changed quickly. The RC map list box contain the placement configuration described in Chapter two. If Custom is selected the file browser is opened and any RC configuration that has been saved can be loaded in the system. The Reserve tick box, box labelled 2, enables the RC configurations to include reserved slots. Altering any of the parameter values updates the two rack systems in real time giving the user an instant reaction to the changes that are made.

APPENDIX D

Results

This chapter contains the table and figures that are not present in Chapter 6. The tables present in this chapter summarises the results obtained from the various simulation replications. Using the summarised data of each the Figures in Chapter 6 are plotted.

D.1 Movement logics analysis (Ratio = 25, 50, 75)

These table are here as their results are only partially used. Only sections of then are referenced and is not needed in the main body of the thesis.

	SSP	OSP	OS	SS	CS
Storage = Retrieval	0	61	49	49	0
Storage > Retrieval	0	22	18	19	1
Storage < Retrieval	1	28	22	21	0

Table D.1: The percentage of the total movement distance, in rack locations, between the best performing movement logic and other for each configuration — ratio of 25 is used.

	SSP	OSP	OS	SS	CS
Storage = Retrieval	0	71	50	50	0
Storage > Retrieval	0	25	18	19	1
Storage < Retrieval	1	33	23	22	0

Table D.2: The percentage of the total horisontal movement distance, in rack locations, between the best performing movement logic and other for each configuration — ratio of 50 is used.

	SSP	OSP	OS	SS	CS
Storage = Retrieval	1	0	43	42	1
Storage > Retrieval	0	0	17	16	0
Storage < Retrieval	0	1	21	20	0

Table D.3: The percentage of the total vertical movement distance, in rack locations, between the best performing movement logic and other for each configuration — ratio of 75 is used.

	SSP	OSP	OS	SS	CS
Storage = Retrieval	84	91	85	85	84
Storage > Retrieval	85	88	85	85	85
Storage < Retrieval	85	88	85	85	85

Table D.4: A summary of the percentage of the total vertical movement distance, in rack locations, between the best performing movement logic and other for each configuration.

	SSP	OSP	OS	SS	CS
Storage = Retrieval	0.25%	0.19%	0.09%	0.15%	0.00%
Storage > Retrieval	0.11%	1.04%	0.39%	0.50%	0.00%
Storage < Retrieval	0.75%	0.43%	0.89%	0.00%	0.50%

Table D.5: A summary of the average total distance, in rack locations, each movement logic took to complete a job list of specific configuration.

D.2 Movement logics analysis (Ratio = 0, 100)

Table D.6 summarises the difference percentage between each movement logic and the best performing movement logic. The result are for when a job list is simulated that only consists of one job type.

	SSP	OSP	OS	SS	CS
Only retrieval	0.49%	0.20%	0.20%	0.00%	0.65%
Only storage	0.39%	0.00%	0.26%	0.15%	0.11%

Table D.6: A summary of the average total distance, in rack locations, each movement logic took to complete a job list of specific configuration.

D.3 Shift 2 analysis

This section contains information used in the analysis of shift 2 (§ 5.2.3). Table D.7 summarises the improvement percentage achieved, for completion time and distance, when a sequencing algorithm is applied. Similarly Table D.8 summarises the improvement percentage achieved, for completion time and distance, when the row slotting algorithm is also applied.

Table D.9 summarises the average travel time, in seconds, it take a high lift to move horizontally and vertically. Travel times are calculated using the triangle distribution function discussed in § 3.5.2. The rows in the table represent the horizontal travel distance, in rack locations, and the columns represent the vertical distance, in racks locations. This means if a high lift was to travel 13 rack locations horizontally and 3 rack locations vertically it would take on average 76.06 seconds.

Table D.10 summarises the average travel distance, in rack locations, when the row slotting algorithm is applied to a almost empty storage rack system. Similarly Table D.11 summarises the travel distance, in rack locations, when the row slotting algorithm is applied to a storage rack system filled 90% to capacity. For both tables the total distance, in rack locations, are split into its **Occupied** and **dead-heading** components. Table D.11

	SSP	OSP	OS	SS	CS
AC	0.25	0.22	2.40	0.68	0.20
AD	0.42	0.41	1.81	0.00	0.46
AE	2.30	2.03	0.07	0.90	1.78
AG	0.54	0.63	0.09	0.81	0.45
AJ	2.14	0.54	1.08	1.02	2.15
AK	0.88	0.77	0.58	1.29	0.24
AL	2.41	2.29	0.88	1.38	3.10
AN	0.37	1.10	0.29	0.62	0.37
AU	0.27	0.34	0.35	1.37	0.20
AV	2.69	1.91	2.40	0.63	3.28
AX	0.41	0.90	0.00	1.13	0.61
AY	1.05	0.83	0.36	1.23	1.53
AZ	4.05	0.71	0.56	0.05	4.64
BA	1.80	1.67	0.50	0.88	2.07
BB	0.17	1.88	0.05	0.30	0.80
BC	1.24	0.25	0.16	0.53	0.76
BD	1.00	2.27	0.68	2.41	1.36
BE	0.57	0.06	0.65	0.49	0.32
BF	0.38	1.39	0.71	0.62	0.12
BG	0.45	0.58	0.81	0.04	0.45
BO	1.20	0.83	0.37	0.56	0.65

(a) Improvement percentages for average completion times, in seconds.

	SSP	OSP	OS	SS	CS
AC	0.00	5.42	1.97	1.03	0.32
AD	12.52	8.43	6.20	5.04	12.47
AE	0.00	11.70	7.98	2.08	0.74
AG	0.00	0.00	0.00	0.00	0.00
AJ	22.19	3.84	3.80	4.97	22.16
AK	15.86	8.93	0.25	7.30	16.38
AL	25.79	16.21	3.74	9.11	25.73
AN	5.10	6.49	0.29	3.06	5.09
AU	4.21	2.34	5.45	9.15	4.19
AV	16.11	14.40	2.69	2.79	16.67
AX	5.77	3.85	0.69	2.26	6.28
AY	14.57	9.37	1.08	0.26	14.23
AZ	12.00	4.66	1.31	5.02	12.43
BA	8.51	0.75	2.71	7.16	8.48
BB	4.71	4.34	0.40	5.24	4.97
BC	1.40	2.08	0.53	3.77	1.26
BD	3.85	8.92	7.06	5.47	4.01
BE	2.59	1.40	1.42	0.56	2.31
BF	18.94	13.17	1.34	5.05	18.77
BG	8.49	0.00	2.18	0.73	9.23
BO	0.00	0.00	0.00	0.00	0.00

(b) Improvement percentages for average completion distance, in rack locations.

Table D.7: Summary of the improvement percentages achieved when applying the sequencer algorithm to the different movement logics for average completion time and distance.

	SSP	OSP	OS	SS	CS
AC	7.58	6.41	5.36	7.52	7.99
AD	4.35	2.92	6.02	8.2	4.21
AE	17.2	9.59	12.07	18.66	17.15
AG	0.45	0.36	0.54	0.9	1.25
AJ	8.46	12.81	16.24	12.03	8.48
AK	6.4	4.91	11.39	7.16	6.65
AL	5.08	5.31	7.75	5.43	5.37
AN	5.5	5.97	14.37	7.71	4.92
AU	8.97	6	8.02	4	4.98
AV	4.32	3.07	1.43	6.83	5.04
AX	3.33	4.25	7.63	5.8	4.27
AY	7.84	3.25	8.88	9.64	7.78
AZ	3.12	3.08	4.47	0.64	3.76
BA	1.74	5.36	10.76	2.26	1.47
BB	1.41	5.76	5.07	2.92	1.47
BC	5.76	2.65	3.26	3	5.09
BD	6.82	2.19	3.95	9.44	7.17
BE	2.9	6.82	7.6	4.94	2.72
BF	3.92	2.34	1.86	5.46	3.09
BG	5	7.99	8	6	6.02
BO	7.02	9	9.01	5.99	9.99

(a) Improvement percentages for average completion times, in seconds.

	SSP	OSP	OS	SS	CS
AC	56.29	47.86	49.34	56.35	56.03
AD	33.66	8.61	39.1	44.26	34.07
AE	76.86	28.67	48.32	74.68	76.85
AG	0	0	0	0	0
AJ	44.38	48.7	59.77	44.72	44.31
AK	25.92	11.33	47.42	39.11	26.01
AL	27.74	26.61	46.94	32.08	28.16
AN	44.13	15.3	61.52	45.12	44.02
AU	25.89	5.76	44.75	37.42	26.45
AV	17.91	16.64	14.7	28.66	18.09
AX	6.3	20.63	35.74	24.48	6.28
AY	30.24	10.77	40.65	39	30.29
AZ	40.36	34.86	47.88	41.61	40.31
BA	8.81	32.53	44.88	22.82	8.79
BB	46.34	45.32	52.52	42.19	46.27
BC	49.44	42.26	50.27	49.56	49.65
BD	52.36	23.08	43.12	55.54	52.36
BE	58.65	55.78	60.66	57.06	58.73
BF	23.24	15.17	29.46	36.35	23.64
BG	45.37	50.08	51.4	44.08	45.77
BO	0	0	0	0	0

(b) Improvement percentages for average completion distance, in rack locations.

Table D.8: Summary of the improvement percentages achieved when applying the sequencer and slotting algorithms to the different movement logics for average completion time and distance.

x/y	0	1	2	3	4	5
1	67.9	69.56	71.26	72.99	74.68	76.37
2	68.17	69.83	71.58	73.21	74.89	76.62
3	68.45	70.11	71.77	73.45	75.2	76.92
4	68.66	70.37	72.05	73.73	75.44	77.16
5	68.97	70.61	72.32	74.04	75.73	77.35
6	69.22	70.91	72.55	74.28	75.93	77.67
7	69.45	71.13	72.82	74.52	76.17	77.94
8	69.68	71.38	73.08	74.81	76.47	78.16
9	70.02	71.64	73.35	75	76.73	78.42
10	70.2	71.88	73.61	75.31	76.97	78.67
11	70.5	72.18	73.81	75.58	77.23	78.89
12	70.72	72.42	74.07	75.81	77.51	79.2
13	70.98	72.66	74.38	76.07	77.71	79.46
14	71.23	72.95	74.6	76.36	78.02	79.69
15	71.51	73.16	74.87	76.59	78.23	79.93
16	71.74	73.46	75.16	76.81	78.58	80.18
17	71.97	73.74	75.4	77.11	78.71	80.46
18	72.28	73.99	75.67	77.3	79.05	80.69
19	72.54	74.21	75.92	77.61	79.3	81
20	72.73	74.5	76.18	77.85	79.53	81.25
21	73.08	74.74	76.38	78.1	79.82	81.52
22	73.31	74.97	76.67	78.4	80.03	81.78
23	73.51	75.24	76.92	78.69	80.32	82.01
24	73.8	75.5	77.24	78.93	80.57	82.29
25	74.07	75.74	77.44	79.14	80.81	82.53
26	74.33	76.02	77.69	79.41	81.09	82.75
27	74.57	76.26	77.98	79.66	81.35	83.05
28	74.86	76.48	78.24	79.9	81.57	83.27
29	75.1	76.78	78.48	80.15	81.84	83.55
30	75.38	77.03	78.68	80.4	82.1	83.78
31	75.61	77.29	78.98	80.66	82.38	84.08
32	75.61	77.29	78.98	80.66	82.38	84.08

Table D.9: Summary of the average travel time, in seconds, a high lifts takes to move from one rack location to another.

	PEP	SSP	OSP	OS	SS	CS
Occupied	82745	52406	52393	52400	52417	52430
Dead-heading	98034	44257	71077	58538	52088	47899
Total	180749	96663	123470	110938	104505	100329

Table D.10: Summary of the cumulative travel distances using row pallet slotting placement algorithm in conjunction with each movement logic split into the occupied, dead-heading and total distance.

	PEP	SSP	OSP	OS	SS	CS
Occupied	82745	67704	67284	66980	66487	68778
Dead-heading	98034	48225	75400	71211	62611	52150
Total	180749	115929	142684	138191	129098	120928

Table D.11: Summary of the cumulative travel distances using row pallet slotting placement algorithm in conjunction with each movement logic split into the occupied, dead-heading and total distance. While simulating each movement logic the storage rack system is filled to 90% capacity.